

A THREE DIMENSIONAL SOLUTION OF THE
TRANSIENT FIELD PROBLEM USING
ISOPARAMETRIC FINITE ELEMENTS

Girard Thomas Lew

Library
Naval Postgraduate School
Monterey, California 93940

NAVAL POSTGRADUATE SCHOOL

Monterey, California



THESIS

A THREE DIMENSIONAL SOLUTION OF THE
TRANSIENT FIELD PROBLEM USING
ISOPARAMETRIC FINITE ELEMENTS

by

Girard Thomas Lew

Thesis Advisors:

D. Salinas
G. Cantin

December 1972

T15-903

Approved for public release; distribution unlimited.

A Three Dimensional Solution of the
Transient Field Problem Using
Isoparametric Finite Elements

by

Girard Thomas Lew
Lieutenant Commander, United States Navy
B.S., United States Naval Academy, 1960

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN MECHANICAL ENGINEERING

from the
NAVAL POSTGRADUATE SCHOOL
December 1972

ABSTRACT

An isoparametric finite element formulation solving a general form of the transient field equation is presented in this thesis. The formulation is developed for fields in three-dimensional Euclidean space. A FORTRAN IV computer program employing double precision arithmetic, compact storage techniques, and providing the option of several numerical integration methods and types of finite elements is presented. Data input may be in rectangular, cylindrical, or spherical coordinates and variations in time integration step size are permitted. Time dependent boundary conditions are not considered.

Comparisons of theoretical and computer solutions for a variety of test problems demonstrate close agreement. Instructions for use of the program are included.

TABLE OF CONTENTS

I.	INTRODUCTION -----	11
II.	FINITE ELEMENT FORMULATION -----	13
	A. ISOPARAMETRIC FINITE ELEMENTS -----	15
	B. MATRIX FORMATION -----	18
	C. BOUNDARY CONDITIONS -----	19
	D. SOLUTION TECHNIQUES -----	20
	1. Systems of Linear Equations -----	21
	2. Steady State -----	21
	3. First Order in Time -----	21
	a. Euler I Method -----	21
	b. Trapezoidal Rule Method -----	22
	c. Linear Time Shape Function Method ---	22
	4. Second Order in Time -----	23
	a. Euler I Method -----	23
	b. Finite Difference Method -----	23
III.	DESCRIPTION OF THE PROGRAM -----	25
	A. ASSUMPTIONS AND RESTRICTIONS -----	25
	B. GENERAL PROGRAM ORGANIZATION -----	30
	C. SUBROUTINES -----	31
	1. Subroutine INPUT -----	31
	2. Subroutine MERGE -----	31
	3. Subroutine BCOND -----	32
	4. Subroutine STEADY -----	32
	5. Subroutine TIME11 -----	32

6.	Subroutine TIME13 -----	33
7.	Subroutine TIME21 -----	33
8.	Subroutine TIME23 -----	33
9.	Subroutine CUBE -----	34
10.	Subroutine FORMC -----	34
11.	Subroutine SHAPE -----	35
12.	Subroutine JACOB -----	35
13.	Subroutine LDLT -----	35
14.	Subroutine MARK -----	36
15.	Subroutine BNSN -----	36
16.	Subroutine MULT -----	36
17.	Subroutine STUFF -----	37
18.	Subroutine PUNC -----	37
IV.	TEST PROBLEMS -----	38
A.	ASSEMBLY TESTS -----	38
B.	STEADY STATE PROBLEMS -----	39
1.	Axial Temperature Distribution in a Cylinder -----	39
2.	Radial Temperature Distribution in a Cylinder -----	41
3.	Chimney Problem -----	45
C.	FIRST ORDER TIME PROBLEMS -----	47
1.	Transient Temperature Distribution in a Hollow Cylinder of Infinite Length ----	47
2.	Transient Heat Conduction in a Prolate Spheroidal Solid -----	53
D.	SECOND ORDER TIME PROBLEMS -----	53
1.	Undamped Membrane -----	55
2.	Damped Membrane -----	55

V. CONCLUSIONS AND RECOMMENDATIONS ----- 58

A. RECOMMENDED MODIFICATIONS TO THE PROGRAM ----- 58

1. Anisotropic Materials ----- 58

2. Time Dependent Boundary Conditions ----- 59

3. Non Linear Material Properties and
Boundary Conditions ----- 59

4. Problem Size ----- 59

5. Specialized Data Input and Output ----- 60

B. RECOMMENDED FUTURE STUDIES ----- 60

APPENDIX A: COMPUTER PROGRAM NOMENCLATURE ----- 61

A. INTEGER CONSTANTS ----- 61

B. REAL CONSTANTS ----- 62

C. VECTORS ----- 62

D. MATRICES ----- 63

APPENDIX B: INSTRUCTIONS FOR PROBLEM DECK
PREPARATION ----- 65

APPENDIX C: STANDARD DECK STRUCTURE ----- 73

APPENDIX D: COMPUTER PROGRAM LISTING ----- 74

LIST OF REFERENCES -----119

INITIAL DISTRIBUTION LIST -----120

FORM DD 1473 -----121

LIST OF TABLES

Table

I.	AXIAL TEMPERATURE DISTRIBUTION IN A CYLINDER ----	40
II.	FINITE ELEMENT MODELS FOR RADIAL CYLINDER PROBLEM -----	42
III.	RADIAL TEMPERATURE DISTRIBUTION IN A HOLLOW CYLINDER WITH THE INTERIOR SURFACE MAINTAINED AT 1500°F -----	43
IV.	RADIAL TEMPERATURE DISTRIBUTION IN A HOLLOW CYLINDER WITH INTERNAL HEAT GENERATION -----	44
V.	TEMPERATURE DISTRIBUTION IN A RECTANGULAR CHIMNEY -----	46
VI.	RADIAL TEMPERATURE DISTRIBUTION IN A CYLINDER AT .05 MINUTES -----	49
VII.	RADIAL TEMPERATURE DISTRIBUTION IN A CYLINDER AT .50 MINUTES -----	50
VIII.	RADIAL TEMPERATURE DISTRIBUTION IN A CYLINDER AT 1.00 MINUTES -----	51
IX.	RADIAL TEMPERATURE DISTRIBUTION IN A CYLINDER AT 1.50 MINUTES -----	52
X.	CENTER NODE DISPLACEMENT OF AN UNDAMPED MEMBRANE -----	56
XI.	CENTER NODE DISPLACEMENT OF A DAMPED MEMBRANE ---	57

LIST OF FIGURES

Figure

1.	Linear Isoparametric Element -----	16
2.	Quadratic Isoparametric Element -----	16
3.	Cubic Isoparametric Element -----	17
4.	Subroutine Calling Sequence for Steady State Problem -----	26
5.	Subroutine Calling Sequence for First Order Time Problem -----	26
6.	Subroutine Calling Sequence for Second Order Time Problem -----	27
7.	Assembly Routines for the Formation of Matrix [H] -----	28
8.	Assembly Routines for the Formation of Matrices [C] and [G] -----	29
9.	Service Routines Used by Solution Routines -----	29
10.	Axial Cylinder Problem Geometry -----	40
11.	Radial Cylinder Problem Geometry -----	43
12.	Rectangular Chimney Problem Geometry -----	46
13.	Transient Temperature Distribution in a Prolate Shperoidal Solid -----	54

LIST OF SYMBOLS AND NOTATION

A. NOTATION

$\langle \rangle$	Row Vector
$[]$	Square Matrix
$\langle \rangle^T$	Transpose of a Row Vector
$[]^T$	Transpose of a Square Matrix
$[]^{-1}$	Inverse of a Square Matrix
$\{ \}$	Column Vector
$[]^e$ or $[]^e$	Matrix or Vector at the Element Level
$\{\phi\}^i$	Vector at i^{th} Point in Time
$\{\dot{\phi}\}$	First Partial Derivative of $\{\phi\}$ with Respect to Time
$\{\ddot{\phi}\}$	Second Partial Derivative of $\{\phi\}$ with Respect to Time
\int_v or \int_{ve}	Volume Integral
\int_s or \int_{se}	Surface Integral
ds	Differential Area in Global Coordinate System
ds_e	Differential Area in Local Coordinate System

B. SYMBOLS

$[C]$ or $[C]^e$	Coefficient Matrix of $\{\dot{\phi}\}$ or $\{\dot{\phi}\}^e$
$\{F\}$ or $\{F\}^e$	Forcing Vector
$[G]$ or $[G]^e$	Coefficient Matrix of $\{\ddot{\phi}\}$ or $\{\ddot{\phi}\}^e$
$[H]$ or $[H]^e$	Coefficient Matrix of $\{\phi\}$ or $\{\phi\}^e$
$[J]$ or $[J_s]$	Jacobian Matrix of Volume or Surface

$\det[J]$ or $\det[J_s]$	Determinant of Jacobian Matrix of Volume or Surface
h	Time Step Size or Heat Transfer Coefficient
k_x, k_y, k_z	Material Property of $[H]$ Matrix in x, y , or z Direction
k	Thermal Conductivity
ℓ_x, ℓ_y, ℓ_z	Direction Cosines
$\langle N \rangle$	Shape Function in Space
$\langle M \rangle$	Shape Function in Time
q	Loss Coefficient per Unit Area or Heat Flux per Unit Area
Q	Generation per Unit Volume or Heat Generation per Unit Volume
r	Radius
r_o	Outside Radius
r_i	Inside Radius
t	Time
T	Temperature
α	Loss Coefficient associated with or Thermal Diffusivity
μ	Material Property of $[C]$ Matrix
ρ	Material Property of $[G]$ Matrix or Density
v	Damping Factor

ACKNOWLEDGEMENTS

The author wishes to express his appreciation to Dr. Gilles Cantin, Professor of Mechanical Engineering, for his advice and guidance throughout the course of development of this text. Without his perseverant assistance this work could not have been accomplished.

The author is obligated to Professor Paul Pucci and Associate Professor Matthew Kelleher for their kind and thoughtful assistance in the preparation of test problems for this work and to Assistant Professor David Salinas for his assistance in the development of time integration procedures.

Finally the author wishes to thank his wife, Annette, for her understanding and encouragement throughout the course of this study.

I. INTRODUCTION

The 'quasi-harmonic' partial differential equation:

$$\begin{aligned} \frac{\partial}{\partial x} \left(k_x \frac{\partial \phi}{\partial x} \right) + \frac{\partial}{\partial y} \left(k_y \frac{\partial \phi}{\partial y} \right) + \frac{\partial}{\partial z} \left(k_z \frac{\partial \phi}{\partial z} \right) \\ + \left(Q - \mu \frac{\partial \phi}{\partial t} - \rho \frac{\partial^2 \phi}{\partial t^2} \right) = 0 \end{aligned} \quad (1)$$

applies to a variety of physical problems [Ref. 10]. The Laplace equation is a typical and well known example of a specialized form of this equation. Since an analytical solution in closed form is seldom available for useful engineering problems, the development of a practical general purpose computerized solution method represents a useful addition to the analytical tools of the engineer.

This thesis describes a general purpose FORTRAN IV computer program using the isoparametric finite element approach to solve the field equation in general form. The program does not presently permit a time variation of boundary conditions or a functional variation of material properties.

The program allows selection of the linear, quadratic, or cubic isoparametric finite element with the option of specifying two through six Gauss points for numerical integration in space. Data input may be a rectangular, cylindrical, or spherical coordinates. Five time integration

routines are provided. Output is presented in a tabular form of node number, space coordinate, and functional value.

The structure of the program is modular in order to accommodate improvements and adjustments due to new developments.

This program was constructed and tested on an IBM-360/67 computer system with OS/360 release 18. Other systems and installations will require modifications.

II. FINITE ELEMENT FORMULATION

The 'quasi-harmonic' or field equation describing the behaviour of some physical quantity, ϕ , is:

$$\begin{aligned} \frac{\partial}{\partial x} \left(k_x \frac{\partial \phi}{\partial x} \right) + \frac{\partial}{\partial y} \left(k_y \frac{\partial \phi}{\partial y} \right) + \frac{\partial}{\partial z} \left(k_z \frac{\partial \phi}{\partial z} \right) \\ + \left(Q - \mu \frac{\partial \phi}{\partial t} - \rho \frac{\partial^2 \phi}{\partial t^2} \right) = 0 \end{aligned} \quad (1)$$

Commonly imposed boundary conditions are of the forms:

$$\phi = \phi_B \quad (2)$$

$$k_x \frac{\partial \phi}{\partial x} l_x + k_y \frac{\partial \phi}{\partial y} l_y + k_z \frac{\partial \phi}{\partial z} l_z + q + \alpha \phi = 0 \quad (3)$$

Where ϕ_B represents the value of ϕ specified on the boundary; l_x , l_y , and l_z are the direction cosines of the outward normal to the boundary surface; q is an outward flux per unit area; Q a generation per unit volume; and k_x , k_y , k_z , μ , and ρ are material properties. Q , k_x , k_y , k_z , μ , and ρ may be functions of space and time.

Reference 10 thoroughly develops the concept of a finite element and the process leading to a discretized form of equation (1) and the boundary conditions given by equations (2) and (3). If the scalar variable, ϕ , is described by a shape function, $\langle N \rangle$, and nodal values of the variable, $\{\phi\}$, such that:

$$\phi = \langle N \rangle \{\phi\} \quad (4)$$

the result is:

$$[H]\{\phi\} + [C]\{\dot{\phi}\} + [G]\{\ddot{\phi}\} + \{F\} = 0 \quad (5)$$

$[H]$, $[C]$, and $[G]$ are symmetric matrices defined, on the element level, as:

$$[C]^e = \int_{V_e} \langle N \rangle^T \mu \langle N \rangle dx dy dz \quad (6)$$

$$[G]^e = \int_{V_e} \langle N \rangle^T \rho \langle N \rangle dx dy dz \quad (7)$$

$$[H]^e = \int_{V_e} \left(k_x \left\langle \frac{\partial N}{\partial x} \right\rangle^T \left\langle \frac{\partial N}{\partial x} \right\rangle + k_y \left\langle \frac{\partial N}{\partial y} \right\rangle \left\langle \frac{\partial N}{\partial y} \right\rangle + k_z \left\langle \frac{\partial N}{\partial z} \right\rangle^T \left\langle \frac{\partial N}{\partial z} \right\rangle \right) dx dy dz \quad (8)$$

The load vector describing boundary conditions is:

$$\begin{aligned} \{F\}^e = & -\int_{V_e} Q \langle N \rangle^T dx dy dz + \int_{S_e} q \langle N \rangle^T ds_e \\ & + \left(\int_{S_e} \langle N \rangle^T \alpha \langle N \rangle ds \right) \{\phi\}^e \end{aligned} \quad (9)$$

where the integrals are performed only over the surface or volume where the prescribed boundary condition applies.

Element matrices and the load vector are assembled by standard methods to form equation (5) which is the discretized form of equation (1). Adding the load vectors given by equation (9) to the rest of the nodal intensities guarantee the satisfaction of the boundary conditions given by equation (3). Finally, the boundary condition of equation (2) can be applied to the assembled set of equations.

The finite element technique models a continuum as an assembly of substructures; thus, for an element of small size, the parameters k_x , k_y , k_z , μ , ρ , α , Q , and q may be considered constant and the integrations required by equations (6), (7), (8), and (9) reduced to an automated numerical technique.

A. ISOPARAMETRIC FINITE ELEMENTS

The linear, quadratic, and cubic isoparametric shape functions are utilized in this development. The resulting elements contain 8, 20, and 32 nodal points respectively. A full development of the concept and properties of this family of finite elements is contained in Ref. 10; only the necessary results will be repeated here.

A "local" non-dimensional system of reference (ξ, η, ζ) is used for convenience of calculation. Results are transformed into the "global" cartesian (x, y, z) system of reference prior to final assembly of problem matrices and vectors. The shape functions, $N_i(\xi, \eta, \zeta)$, for the element types used in this development are given in Ref. 10 and need not be repeated here. The nodal numbering conventions employed for each element type are shown in Figures 1, 2, and 3.

Important results relating the shape function to the local and global reference systems [Ref. 10] are:

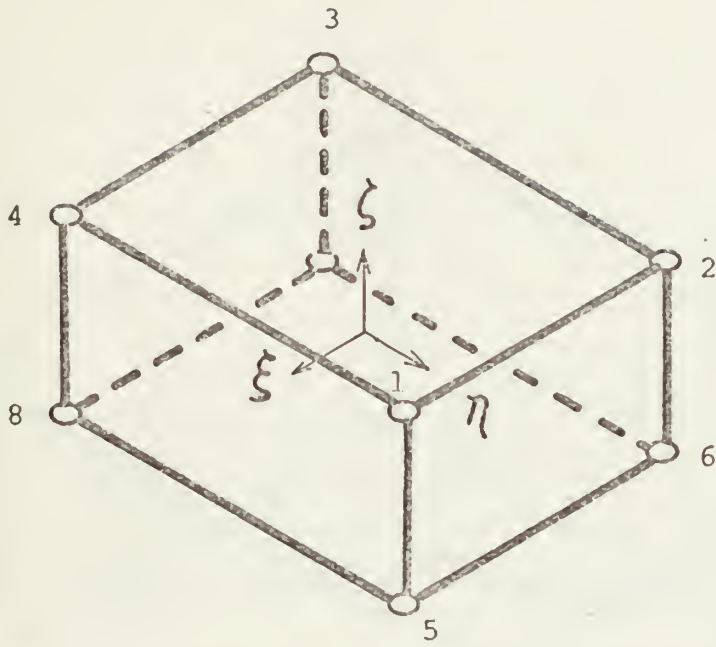


Figure 1. Linear Isoparametric Element.

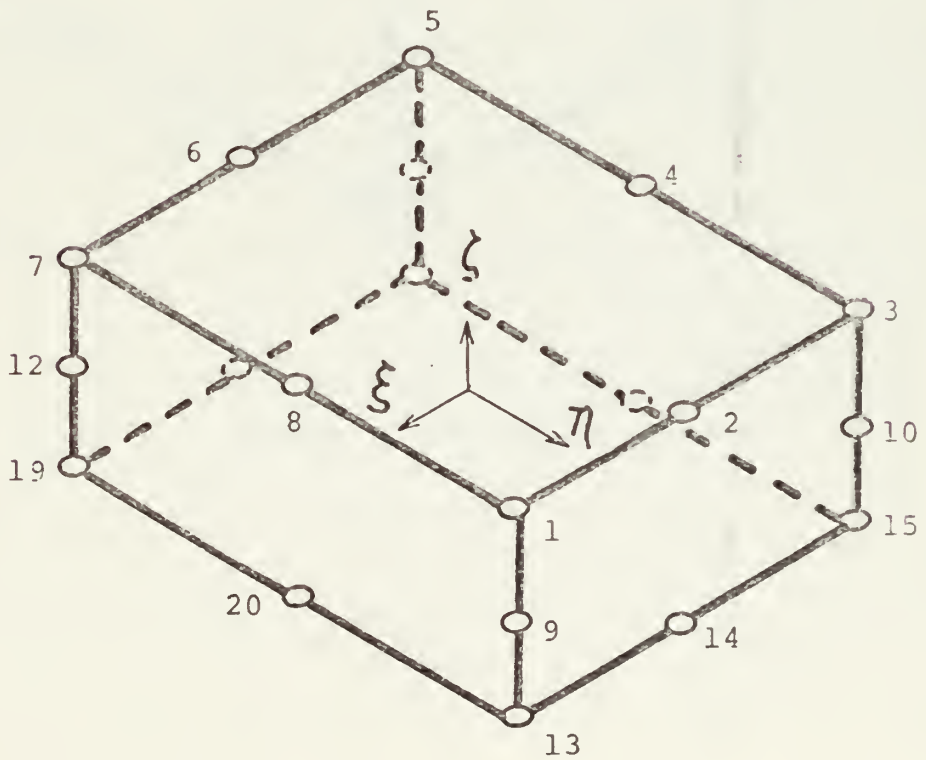


Figure 2. Quadratic Isoparametric Element.

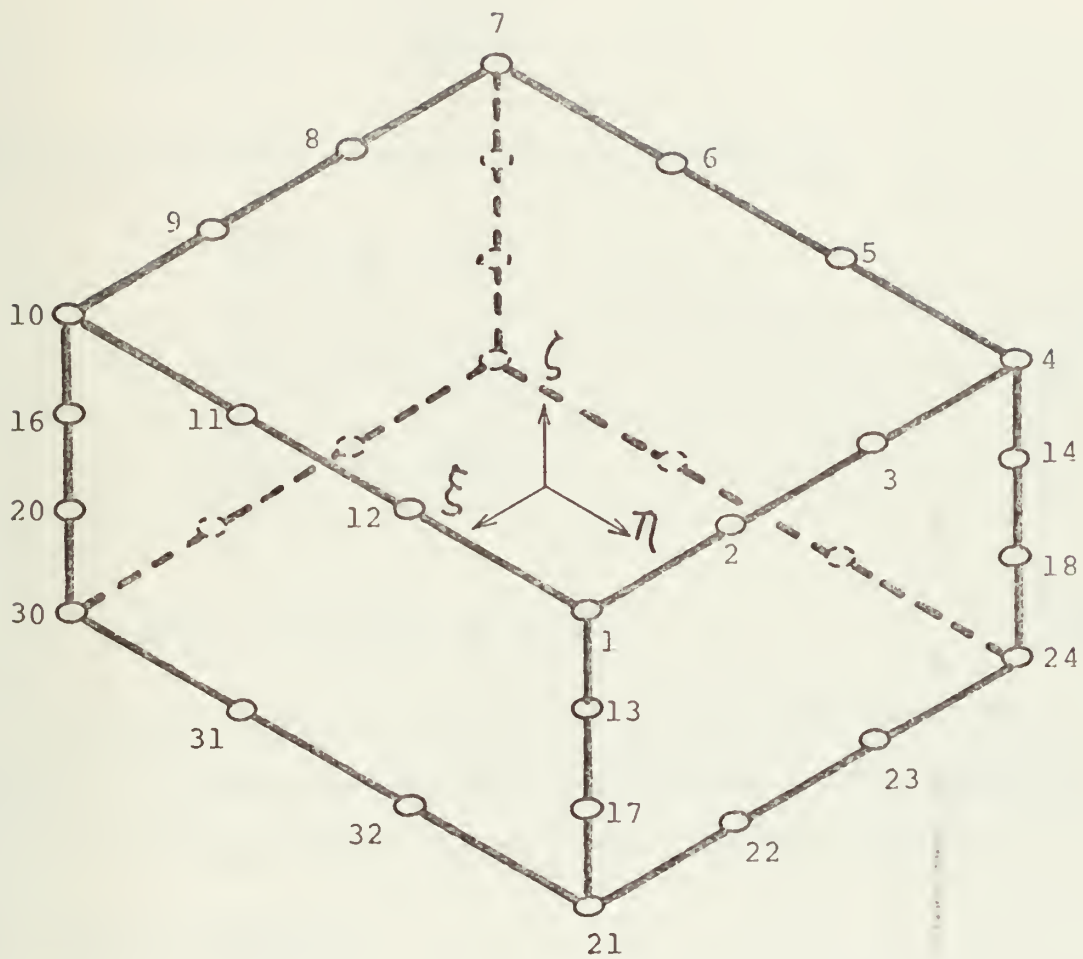


Figure 3. Cubic Isoparametric Element.

$$\begin{Bmatrix} \frac{\partial N_i}{\partial \xi} \\ N_i \\ \frac{\partial N_i}{\partial \eta} \\ \frac{\partial N_i}{\partial \zeta} \end{Bmatrix} = [J] \begin{Bmatrix} \frac{\partial N_i}{\partial x} \\ \frac{\partial N_i}{\partial y} \\ \frac{\partial N_i}{\partial z} \end{Bmatrix} \quad (10)$$

where the Jacobian matrix, $[J]$, is defined as

$$[J] = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} & \frac{\partial z}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} & \frac{\partial z}{\partial \eta} \\ \frac{\partial x}{\partial \zeta} & \frac{\partial y}{\partial \zeta} & \frac{\partial z}{\partial \zeta} \end{bmatrix} \quad (11)$$

and

$$dx dy dz = \det [J] d\xi d\eta d\zeta \quad (12)$$

The partial derivatives of the shape function with respect to global coordinates are recovered from equation (10) by applying the inverse of the Jacobian matrix.

B. MATRIX FORMATION

Assuming constant material properties within an element and applying equations (10), (11), and (12) to the matrix and vector formulations of equations (6), (7), (8) and (9) reduces the form of the integrals to:

$$\int \langle N \rangle^T \langle N \rangle \det [J] d\xi d\eta d\zeta$$

$$\int \left\langle \frac{\partial N}{\partial x} \right\rangle^T \left\langle \frac{\partial N}{\partial x} \right\rangle \det [J] d\xi d\eta d\zeta + \dots$$

$$\int \langle N \rangle^T \det [J] d\xi d\eta d\zeta$$

$$\int \langle N \rangle^T \det [J_s] ds_e$$

$$\int \langle N \rangle^T \langle N \rangle \det [J_s] ds_e$$

where the integrations are performed in the local nondimensional coordinate system. The forms of equations (6), (7), (8), and (9) become ideally suited for a Gaussian numerical integration procedure.

C. BOUNDARY CONDITIONS

The specification of a value of the function ϕ along a given boundary as in equation (2) is straightforward and requires no comment.

Boundary specifications of equation (3) appear in the expression for a forcing vector given by equation (9). The first term of equation (9) containing Q represents a generation rate per unit volume; the second term containing q represents a flux per unit area; and the third term containing α represents a surface loss coefficient per unit area. The terms containing q and Q contribute to a forcing vector, $\{F\}$, while the term containing α results in a correction to the corresponding element $[H]$ matrix.

If, in equation (5), q and α are zero and k_x , k_y , and k_z are equal the boundary condition reduces to the requirement that the normal derivative be equal to zero. This condition is automatically imposed by equation (5) in the absence of any other boundary condition specification.

Arpaci [Ref. 1] gives a number of boundary conditions applicable to the heat problem. Of these the Radiation,

Interface in Motion, and Change of Phase are not included in this development.

Consider an element located at the surface of a body in space. From Figures 1, 2, and 3 observe that a corner node may be common to one, two, or three boundary surfaces per element and that a mid side node is common to one or two boundary surfaces per element. Each boundary surface may be subject to a different boundary condition specification. If two or more of the element surfaces have different boundary conditions imposed, a unique situation arises if the boundary specifications include conditions on ϕ and q or α . In this case the boundary conditions must be specified in order. First, all boundary conditions involving q and α are formed and merged into the overall system of equations; then the boundary conditions involving ϕ are applied.

D. SOLUTION TECHNIQUES

For a fixed point in time and for steady state problems, equation (5) reduces to the form

$$[A]\{\phi\} = \{B\} \quad (13)$$

where $[A]$ is a $n \times n$ square matrix. The vector $\{B\}$ is fully determined in all cases except where a nodal value of ϕ is specified. If a nodal value, ϕ_i , is specified the corresponding forcing vector term, B_i , will contain the unknown quantity of the equation.

1. Systems of Linear Equations

Equation (13) may be solved by standard methods of linear algebra. The method employed here to sort the known and unknown quantities of equation (13) was developed by Irons [Ref. 6] and has the advantages of preserving the symmetry of matrix [A] and permitting the determination of all unknown quantities of the system of equations.

After applying the Irons [Ref. 6] modifications, solution of equation (13) is accomplished by the reduction of matrix [A] into the product of lower unit triangular, diagonal, and upper triangular matrices such that:

$$[A] = [L] [D] [L]^T$$

Unknown quantities are found by performing a forward substitution followed by a backward substitution [Ref. 13].

2. Steady State

In steady state, equation (5) readily reduces to the form of equation (13) and is solved by the technique described previously.

3. First Order in Time

At fixed points in time, equation (5) reduces to the form of equation (13). The numerical integration schemes employed are straightforward and are summarized below. The initial value of $\{\phi\}$ is assumed known and time step size is denoted by h .

a. Euler I Method

This integration scheme is well documented by Crandall [Ref. 3]. The result, for the i^{th} point in time, is:

$$[C]\{\dot{\phi}\}^{i-1} = -[H]\{\phi\}^{i-1} - \{F\} \quad (15a)$$

$$\{\phi\}^i = \{\phi\}^{i-1} + h \{\dot{\phi}\}^{i-1} \quad (15b)$$

b. Trapezoidal Rule Method

This approach to time integration is documented by Ref. 3 and Ref. 11. At mid interval

$$\{\phi\}^{i+\frac{1}{2}} = \frac{\{\phi\}^i + \{\phi\}^{i+1}}{2}$$

$$\{\dot{\phi}\}^{i+\frac{1}{2}} = \frac{\{\phi\}^{i+1} - \{\phi\}^i}{h}$$

When applied to equation (5), the resulting recursion formula is:

$$([H] + \frac{1}{h} [C])\{\phi\}^{i+1} = (-[H] + \frac{1}{h} [C])\{\phi\}^i - 2\{F\} \quad (16)$$

c. Linear Time Shape Function Method

Reference 10 develops the following approach to time integration. Assume that $\{\phi\}$ may be considered as a product of the nodal values of $\{\phi\}$ at discrete times and a time shape function, $M(t)$, such that

$$\langle M \rangle = \langle M_0(t), M_1(t) \rangle$$

where, for a linear variation:

$$M_0 = \frac{(h-t)}{h}$$

and

$$M_1 = t/h$$

the result is:

$$\{\phi\} = \langle M_0, M_1 \rangle \begin{Bmatrix} \{\phi\}^{i-1} \\ \{\phi\}^i \end{Bmatrix}$$

Applying this relationship and the Galerkin process described in Ref. 10 results in the formula:

$$\left(\frac{2}{3} [H] + \frac{1}{h} [C]\right)\{\phi\}^i = - \left(\frac{1}{3} [H] - \frac{1}{h} [C]\right)\{\phi\}^{i-1} - \{F\} \quad (17)$$

4. Second Order in Time

At fixed points in time, equation (5) reduces to the form of equation (13). The integration methods employed are summarized below. Initial values of $\{\phi\}$ and $\{\dot{\phi}\}$ are assumed to be known. Time step size is denoted by h .

a. Euler I Method

The Euler I integration method is well known and documented by many standard sources such as Ref. 3. The result, when applied to equation (5) is:

$$[G]\{\ddot{\phi}\}^{i-1} = -[H]\{\phi\}^{i-1} - [C]\{\dot{\phi}\}^{i-1} - \{F\} \quad (18a)$$

$$\{\phi\}^i = \{\phi\}^{i-1} + h \{\dot{\phi}\}^{i-1} \quad (18b)$$

$$\{\dot{\phi}\}^i = \{\dot{\phi}\}^{i-1} + h \{\ddot{\phi}\}^{i-1} \quad (18c)$$

In the event that the $[C]$ matrix equals zero these equations remain valid.

b. Finite Difference Method

Finite difference approximations are developed in Ref. 3. A four point backward difference approximation for $\{\ddot{\phi}\}$ and a three point backward difference approximation



for $\{\dot{\phi}\}$ have truncation errors of the same order of magnitude, h^2 , and are:

$$\{\ddot{\phi}\}^i = \frac{1}{h^2} (-\{\phi\}^{i-3} + 4\{\phi\}^{i-2} - 5\{\phi\}^{i-1} + 2\{\phi\}^i) \quad (19a)$$

$$\{\dot{\phi}\}^i = \frac{1}{h} \left(\frac{1}{2} \{\phi\}^{i-2} - 2\{\phi\}^{i-1} + \frac{2}{3} \{\phi\}^i \right) \quad (19b)$$

Define

$$\{\phi^*\} = \frac{1}{h^2} (-\{\phi\}^{i-3} + 4\{\phi\}^{i-2} - 5\{\phi\}^{i-1}) \quad (19c)$$

and

$$\{\phi^{**}\} = \frac{1}{h} \left(\frac{1}{2} \{\phi\}^{i-2} - 2\{\phi\}^{i-1} \right) \quad (19d)$$

Equations (19a) and (19b) when applied to equation (5) give the following recursion formula

$$\begin{aligned} ([H] + \frac{2}{h^2} [G] + \frac{3}{2h} [C]) \{\phi\}^i = \\ - \{F\} - [G]\{\phi^*\} - [C]\{\phi^{**}\} \end{aligned} \quad (20)$$

if $[C]$ is not equal to zero and

$$([H] + \frac{2}{h^2} [G]) \{\phi\}^i = - \{F\} - [G]\{\phi^*\} \quad (21)$$

if $[C]$ equals zero.

Equation (21) is known as the Houbolt method and is shown by Johnson [Ref. 7] to be unconditionally stable.

Three starting values of $\{\phi\}$ must be obtained by independent means such as the application of the Euler I method or a Taylor series expansion.

III. DESCRIPTION OF THE PROGRAM

A functional block diagram of the program is presented in Figures 4 through 9. A modular structure was adopted in order to permit simple modifications to accommodate future improvements and the adaptation of the program for special purposes.

The following paragraphs give a brief description of the function and operation of the program. Appendices A through D discuss the preparation of input data, list the important computer program nomenclature, and list the program. A full understanding of the operation of this program can only be achieved by a study of Appendices A through D and an understanding of the principles and solution techniques employed.

A. ASSUMPTIONS AND RESTRICTIONS

The key assumptions and restrictions which impose restrictions in the type of problems that can be solved by the program are:

1. Material properties k_x , k_y , k_z , ρ , μ , and α and problem parameters q and Q are constant functions of time and space throughout the element and are independent of the function ϕ .
2. Boundary values of ϕ are not functions of time.
3. The principal axes of each element align with the principal axes of the global coordinate system and are fixed

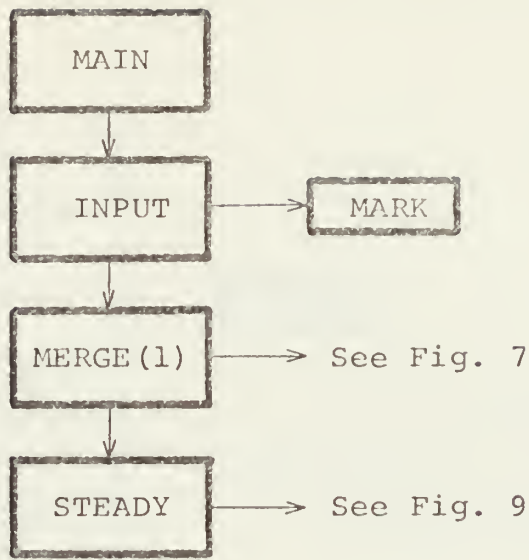


Figure 4. Subroutine Calling Sequence for Steady State Problem.

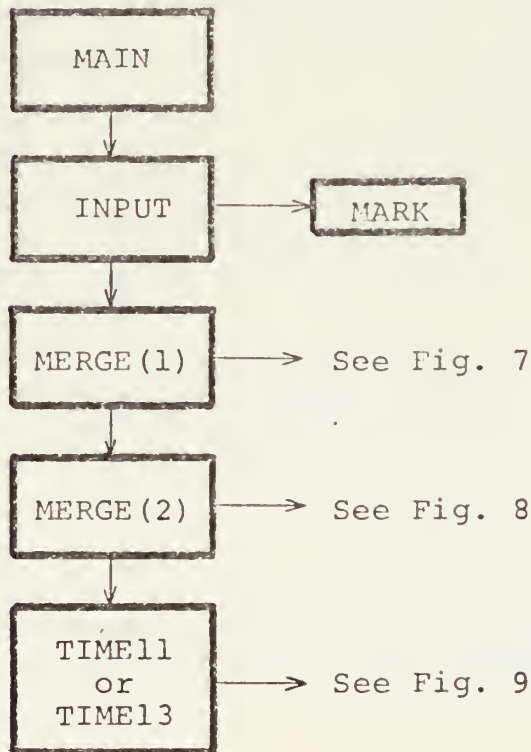
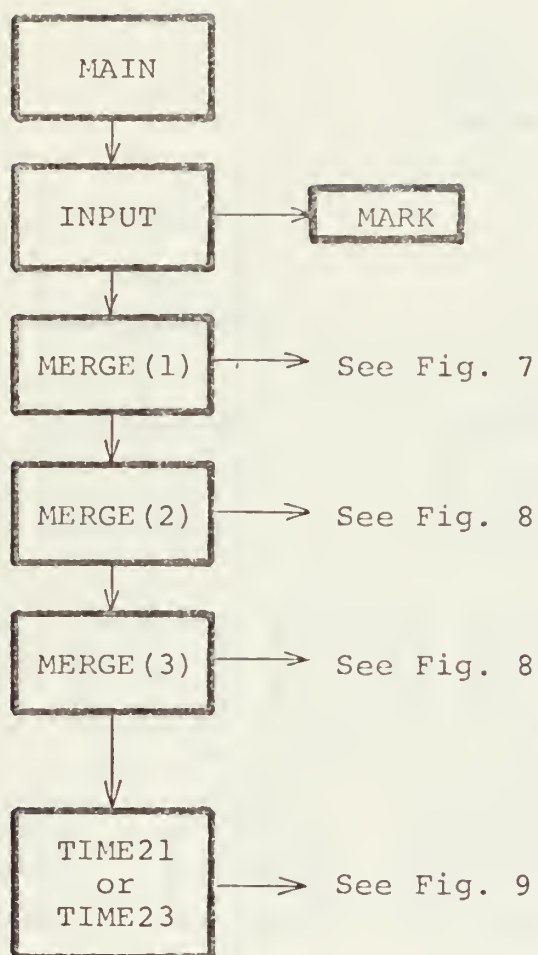


Figure 5. Subroutine Calling Sequence for First Order Time Problem.



NOTE: MERGE(2) is not called if the [C] matrix is absent

Figure 6. Subroutine Calling Sequence for Second Order Time Problem.

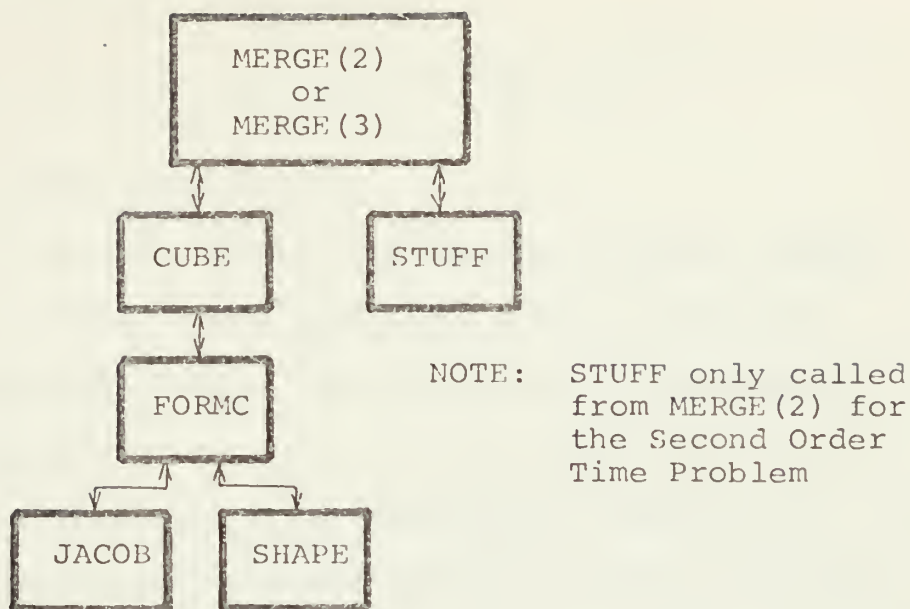
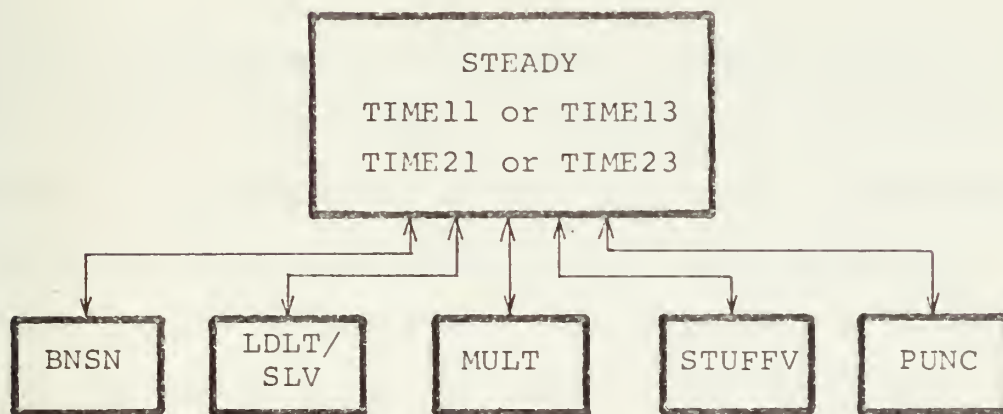


Figure 8. Assembly Routines for the Formation of Matrices [C] and [G].



NOTE: (1) STUFF called by TIME23 only
 (2) MULT not called by STEADY

Figure 9. Service Routines Used by Solution Routines.



in direction throughout the element.

These limitations will be discussed in Section V.

B. GENERAL PROGRAM ORGANIZATION

The program may be divided into the three major independent functions: reading of input data, matrix formation, and actual problem solution. This organization is shown in Figures 4 through 9.

All matrices involved in the problem are symmetrical. At the element level the complete matrix is formed; however, formation of the master matrices $[H]$, $[C]$, and $[G]$ and problem solution utilize a standard compressed storage mode to conserve computer core. The technique consists of placing the diagonal elements of each row of the matrix into the first column and retaining only the upper elements of the matrix.

Reading of necessary input data is accomplished by subroutine INPUT. Preparation of input data is discussed in Appendix B. An echo check of all input data is provided.

Formation of the required matrices and vectors is controlled by subroutine MERGE. Matrices $[H]$, $[C]$, and $[G]$ are formed sequentially. This permits output of the completed matrix to an external storage device and conserves core storage. Boundary condition calculations are performed during formation of the $[H]$ matrix. Associated subroutines include CUBE, FORM, and JACOB.

The transient solution subroutines are similar in construction and differ only in the integration procedure employed. Time integration step size changes and data output are controlled by the solution subroutines. Certain functions common to all solution subroutines are performed by a set of service subroutines which include LDLT, BNSN, STUFF, PUNC, and MULT.

C. SUBROUTINES

The MAIN program is simply a control segment that serves to call INPUT; MERGE(1), MERGE(2), or MERGE(3) in the proper order and as required; and the applicable solution subroutine STEADY, TIME11, TIME13, TIME21, or TIME23. No calculations are performed in the MAIN program

1. Subroutine INPUT

This subroutine reads all input data and provides the appropriate echo check. The number of nodes per element is determined from the element type specification and, if required, cylindrical or spherical coordinates are transformed into rectangular coordinates by calling subroutine MARK.

2. Subroutine MERGE(IMTRX)

This subroutine forms the master matrix specified by the calling argument, IMTRX, by assembling each element matrix into the master matrix. A call to subroutine CUBE returns the desired element matrix. IMTRX=1 corresponds to the [H] matrix, IMTRX=2 corresponds to the [C] matrix, and

IMTRX=3 corresponds to the [G] matrix. Subroutine BCOND is called during formation of the [H] matrix and, if required by the solution subroutine selected, the [H] or [C] matrices are placed into the appropriate storage location after formation.

3. Subroutine BCOND

This subroutine forms the element vectors and matrices required by the appropriate boundary condition specified and assembles the resulting vector or modification matrix into the forcing vector, {F}, or the corresponding [H] matrix. Formation at the element level is accomplished by a call to the appropriate entry point of subroutine CUBE.

4. Subroutine STEADY

This subroutine solves the steady state problem. After the application of appropriate nodal boundary conditions, a call to subroutine LDLT and entry point SLV produces the solution. Data output is a printout of the $\{\phi\}$ and {F} vectors and punched card output of the $\{\phi\}$ vector if requested.

5. Subroutine TIME11

This subroutine employs the Euler I time integration method, equations (15a) and (15b), to solve the first order time problem. One call to subroutine LDLT is required; subsequent calls to entry point SLV produce the desired solution. Data output is a tabular presentation of the vector at specified points in time with punched card output of the $\{\phi\}$ vector provided if requested.

6. Subroutine TIME13

This subroutine employs the "Linear Time Shape Function" as the time integration method (KINT=3), equation (17), or the Trapezoidal Rule time integration method (KINT=2), equation (16), to solve the first order time problem. External core storage devices are utilized to store the [H] and [C] matrices in order to permit restarting the problem at each change in time integration step size. One call to subroutine LDLT is required for each change in time integration step size with the required solution obtained by sequential calls to entry point SLV. Data output is the same as for TIME11.

7. Subroutine TIME21

This subroutine employs the Euler I time integration method, equations (18a), (18b), and (18c), to solve the second order time problem. Operation of this subroutine is identical to the operation of TIME11 except for the inclusion of the vector $\{\ddot{\phi}\}$ in the data output.

8. Subroutine TIME23

This subroutine employs a Four Point Backward Finite Difference (Houbolt) time integration method, equations (20) and (21), to solve the second order time problem. Starting values are obtained by a five term Taylor Series expansion. Operation of this subroutine is similar to the operation of subroutine TIME13 employing external core storage devices and saving the values of $\{\ddot{\phi}\}$ required to restart the problem after a change in time integration step size. Data output is the same as for TIME21.

9. Subroutine CUBE(NPEL,NGP,ITYPE)

This subroutine performs the Gaussian numerical integration required for the formation of the various element matrices. Entry arguments are the number of nodes per element, number of Gauss points specified for the integration process, and the type of integration to be performed. Gauss points and weighting factors were obtained from Ref. 9. A selection of two through six Gauss points with a default option of three Gauss points is provided. For each specified Gaussian integration point, the numerical evaluation of the integrand is determined by calling subroutine FORMC at the appropriate entry point and applying the proper weight factor to the result prior to summation.

The specialized volume and surface integrals required for boundary condition calculations are obtained by entering this subroutine at entry point CUBEB.

10. Subroutine FORMC(XI,ETA,ZETA,NPEL)

This subroutine performs the products of $\langle N \rangle^T \langle N \rangle$, $\langle \frac{\partial N}{\partial x} \rangle^T \langle \frac{\partial N}{\partial x} \rangle$, etc. required at each point in local coordinate space for the numerical integration process. Values of the shape function and partial derivatives of the shape function with respect to the local coordinate system are determined by a call to subroutine SHAPE. Calling subroutine JACOB returns the required values of $[J]$, $[J]^{-1}$, and $\text{Det}[J]$. Necessary entry arguments are the local coordinates of the point in space under consideration and the number of points per element. Various entry points are employed to accommodate

the specialized calculations required in the formation of each element matrix or vector and volume or area integrations.

11. Subroutine SHAPE(XI,ETA,ZETA,NPEL)

This subroutine forms the shape function and the partial derivative of the shape function with respect to the local coordinate system at the point in local coordinate space specified by the entry argument. The type of element to be considered is determined by the specification of the number of points per element in the calling argument.

12. Subroutine JACOB(NPEL,AJ,DCOL,CORD,AJI,DTJ)

This subroutine evaluates the Jacobian matrix, determines the inverse of the Jacobian matrix and the determinant of the Jacobian. The number of points per element, global coordinates of the node points, and shape function partial derivatives with respect to the local system of reference are required entry arguments. $[J]$ is returned in AJ, $[J]^{-1}$ in AJI, and $\det[J]$ in DTJ.

The specialized calculations required for a surface integration are accomplished by entering this subroutine at entry point JACOB.

13. Subroutine LDLT(A,N,M,ASN)

This subroutine reduces the matrix $[A]$ into the products of lower unit triangular, diagonal, and upper triangular matrices, $[L] [D] [L]^T$, required by the process of solving the linear system of equations described by $[A] \{x\} = \{B\}$. N and M are the number of equations and half band width respectively. ASN is an arbitrarily small number;

numbers less than ASN squared are considered to be zero. Entry point SLV(Z,B,N,M) performs the forward substitution and backward substitution required to solve the system of equations.

14. Subroutine MARK(C,K,M)

This subroutine converts coordinates given in a cylindrical or spherical system of reference to a rectangular system of reference. C is a matrix containing the specified coordinates, K is a code to indicate the system of reference employed, and M is the number of coordinates specified.

15. Subroutine BNSN(A,N,APZRO,ABIGN)

This subroutine determines the arbitrarily small number required by subroutine LDLT and the arbitrarily large number required by the equation solving process. APZRO is determined from the matrix [A] by finding the average value of the diagonal elements and dividing the result by 10^{15} . ABIGN is $10^{30} \times \text{ASN}$. This provides the numerical spread necessary for the IBM/360 computer installation employed for this program development.

16. Subroutine MULT(A,B,C,N,M)

This subroutine performs the matrix multiplication $[A] \{B\} = \{C\}$ in a compressed storage form. This multiplication is required by the various time integration subroutines. N and M are the number of equations and half band width respectively.

17. Subroutine STUFF(A,B,N,M)

This subroutine places the matrix [A] into matrix [B]; or entry STUFFV(AA,BB,N,MC) places the vector {B} into the specified column, MC, of matrix [AA].

18. Subroutine PUNC(A,B,N,K)

This subroutine produces the punched output requested by the solution subroutines STEADY, TIME11, TIME13, TIME21, or TIME23.

IV. TEST PROBLEMS

The test problems employed here are designed to test and verify the operation of the computer program. The four categories of tests performed were assembly, steady state, first order in time, and second order in time problems.

The assembly tests served to verify the calculations and integrations required to form the problem matrices and vectors on the element level. Steady state problems were used to verify the matrix assembly process, linear equation solution process, and the validity of boundary condition formulations. Transient problems of first and second order in time served to test the various solution subroutines utilized.

A. ASSEMBLY TESTS

A rectangular region in space provides a convenient means of verifying element matrix and vector assembly procedures. The calculation of all problem matrices and vectors is straightforward and element calculations of the $[H]$, $[C]$, and $[G]$ matrices were verified in this manner. Similarly, the basic assembly processes of subroutines MERGE and BCOND are readily verified.

The integrations required by equations (6), (7), (8) and (9) are performed by a Gaussian numerical integration scheme. In the Gauss procedure, n sampling points will exactly integrate a polynomial of degree $(2n-1)$. The integrations necessary may involve polynomials of degree 2 through 14 in

any given direction of the local coordinate system. Polynomial degree will depend on the shape of the element selected. For practical applications, a selection of two through six Gauss points is sufficient. For a rectangular shape the linear quadratic, and cubic elements are exactly integrated by 2, 3, and 4 Gauss points respectively. Verification of the number of Gauss points required to exactly integrate elements of other shapes was not considered.

3. STEADY STATE PROBLEMS

1. Axial Temperature Distribution in a Cylinder

Consider a cylinder of the geometry specified in Figure 10. If the sides are perfectly insulated a linear temperature distribution results when end conditions are specified [Refs. 1 and 5]. Specify the thermal conductivity, k , as 20 BTU/hr/ft/°F and divide the region into two finite elements. The top surface is subjected to an ambient temperature of 100°F.

If the bottom of the cylinder is subjected to a heat flux, q , of 500 BTU/hr/ft² or maintained at 300°F with a heat transfer coefficient, α , of 5 BTU/hr/ft²/°F identical temperature distributions result. Theoretical and finite element solutions are shown in Table I.

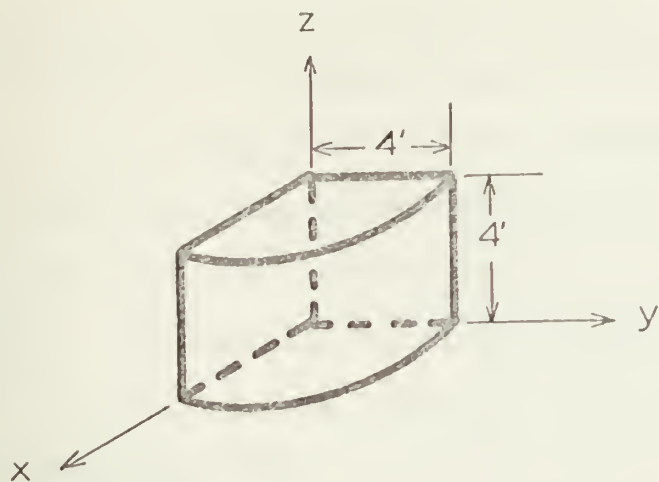


Figure 10. Axial Cylinder Problem Geometry.

TABLE I
AXIAL TEMPERATURE DISTRIBUTION IN A CYLINDER

Distance from Bottom(ft)	Temperature (°F)		
	Theoretical Solution	2 Quadratic Element Model	2 Cubic Element Model
0.00	200.00	200.00	200.00
0.67	183.33		183.33
1.00	175.00	175.00	
1.33	166.67		166.67
2.00	150.00	150.00	150.00
2.67	133.33		133.33
3.00	125.00	125.00	
3.33	116.67		116.67
4.00	100.00	100.00	100.00

2. Radial Temperature Distribution in a Cylinder

Consider a hollow cylinder of infinite length. A segment of such a cylinder is shown in Figure 11. Heat is lost by convection through the outer wall. Consider the case of the interior surface maintained at a constant temperature or the case of a specified heat generation within the cylinder with the interior surface perfectly insulated. Assumed problem parameters are:

a. Inside radius (r_i)	3 in.
b. Outside radius (r_o)	9 in.
c. Heat transfer coefficient (α)	36 BTU/hr/ft ² /°F
d. Thermal conductivity (k)	6 BTU/hr/ft/°F
e. Ambient temperature	70°F
f. Inside wall temperature	1500 °F
g. Heat generation rate (Q)	432 BTU/hr/ft ³

Theoretical solutions to this problem are readily obtained by methods outlined in Refs. 1 and 5. Comparisons of solutions obtained by finite element models and theoretical means are given in Tables III and IV.

A finite element model of this problem is constructed by assembling elements radially to form a segment of the cylinder. Since the problem is symmetric and no heat is conducted perpendicular to any radius, any convenient specification of element arc width, θ , and axial thickness, L , may be used. A small element arc is desirable since the isoparametric finite element used will provide a better approximation to a true circular arc. A large arc width

will introduce some distortion in the temperature distribution. Note that the sum of the forcing vector elements represents the total heat flux through the body in this case. The various finite element models used are given in Table II.

TABLE II
FINITE ELEMENT MODELS FOR RADIAL CYLINDER PROBLEM

	Model			
	L-12	L-24	Q-6	C-4
Element Thickness (L,in)	1	1	1.5	1.5
Element Width (θ , $^{\circ}$)	2	2	6	6
Element Type	Linear	Linear	Quadratic	Cubic
Number of Elements	12	24	6	4
Number of Nodes	52	100	80	92

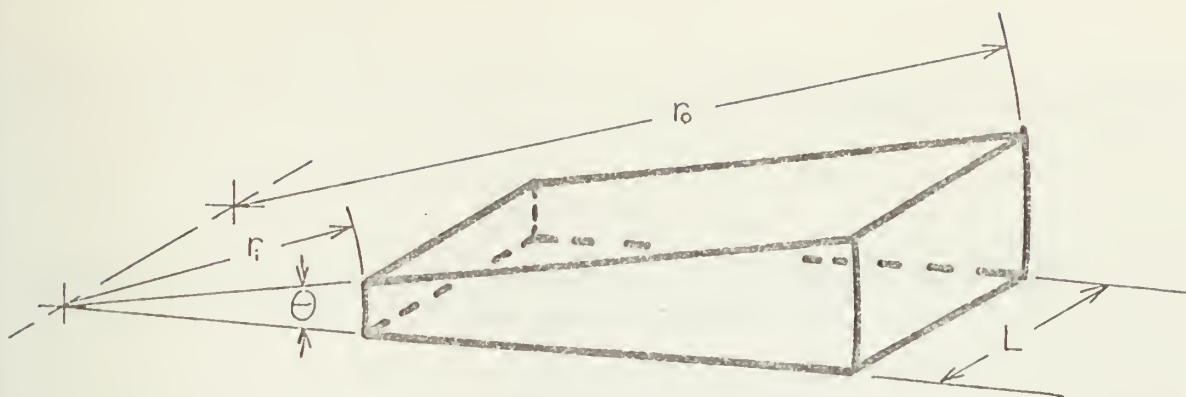


Figure 11. Radial Cylinder Problem Geometry.

TABLE III

RADIAL TEMPERATURE DISTRIBUTION IN A HOLLOW CYLINDER
WITH THE INTERIOR SURFACE MAINTAINED AT 1500 °F

Radius (in)	Temperature (°F)				
	Theoretical	Model L-12	Model L-24	Model Q-6	Model C-4
3	1500.00	1500.00	1500.00	1500.00	1500.00
4	1188.54	1188.85	1188.63	1188.57	1188.53
5	946.96	947.34	947.06	947.01	946.99
6	749.56	749.94	749.67	749.63	749.62
7	582.67	583.00	582.77	582.75	582.74
8	438.11	438.38	438.20	438.19	438.19
9	310.59	310.81	310.67	310.69	310.68
Heat Flux (q)					
Theoretical		9.45	9.45	85.03	85.03
Calculated		9.46	9.46	85.03	85.02

TABLE IV

RADIAL TEMPERATURE DISTRIBUTION IN A HOLLOW CYLINDER
WITH INTERNAL HEAT GENERATION

Radius (in)	Temperature (°F)			
	Theoretical	Model L-12	Model Q-6	Model C-4
3	175.28	175.34	175.30	175.30
4	173.00	173.03	173.02	173.02
5	166.77	166.79	166.79	166.79
6	157.13	157.13	157.15	157.14
7	144.35	144.34	144.36	144.36
8	128.60	128.60	128.62	128.62
9	110.00	109.99	110.02	110.02
Heat Flux (q)				
Theoretical		1.57	14.14	14.14
Calculated		1.57	14.14	14.14

3. Chimney Problem

Consider the two dimensional heat transfer problem posed by a rectangular chimney with the inside surface maintained at 100°F and the outside surface maintained at 0°F . The steady state temperature distribution may be obtained by constructing a finite element model consisting of the desired number and type of elements assembled to form a plate having the required geometry. The geometry of such a model utilizing symmetry boundary conditions is shown in Figure 12 and the results of various finite element models are compared with a solution obtained by a relaxation technique in Table V at the points shown in Figure 12.

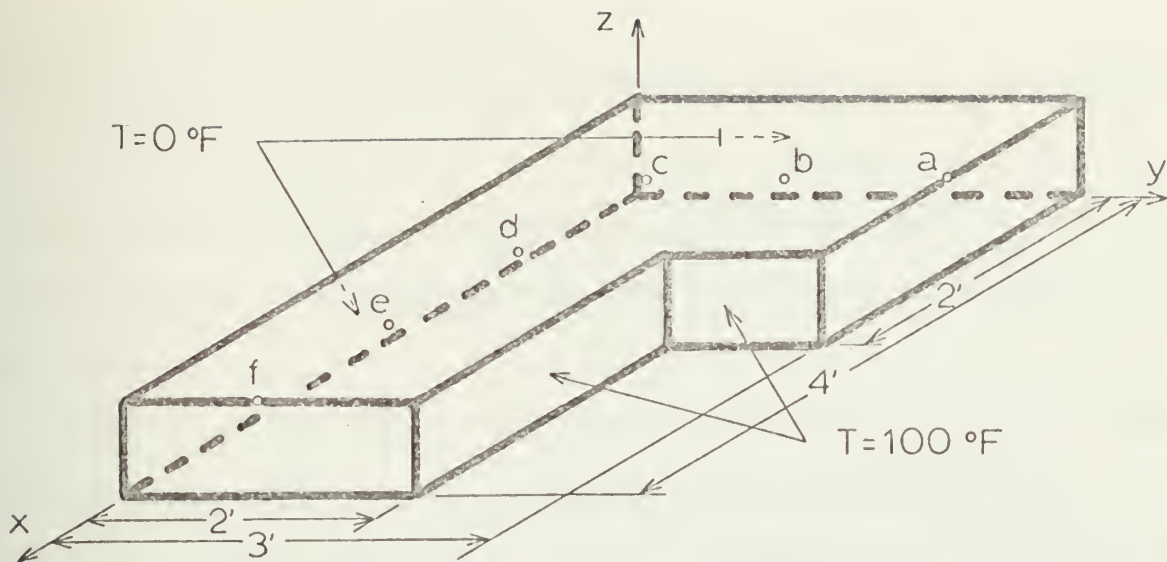


Figure 12. Rectangular Chimney Problem Geometry.

TABLE V
TEMPERATURE DISTRIBUTION IN A RECTANGULAR CHIMNEY

Point	(x,y)	Relaxation Method	Temperature (°F)		
			5 Cubic Element Model	10 Quadratic Element Model	40 Linear Element Model
a	1,3	45.5		44.3	45.2
b	1,2	38.9		37.5	38.3
c	1,1	20.6		19.4	20.3
d	2,1	39.2	37.1	37.9	38.7
e	3,1	47.7		47.0	47.5
f	4,1	49.3	48.7	48.9	49.1

C. FIRST ORDER TIME PROBLEMS

Transient heat conduction in various bodies is a common form of the field equation that is easily modeled by the finite element process. Transient heat conduction in a hollow cylinder and a prolate spheroid will be considered here.

Euler I, Trapezoidal, and Linear Time Shape Function time integration methods were employed. Each method yielded excellent results when used with an appropriate time step size. However, with the cylinder problem, the Euler I method was unstable for large time increments. The Trapezoidal and Linear Time Shape Function methods produced excellent results in all cases.

1. Transient Temperature Distribution in a Hollow Cylinder of Infinite Length

Consider the hollow cylinder of Steady State Test 2, Figure 11. Thermal diffusivity, α , is specified as $.25 \text{ ft}^2/\text{hr}$ where $\alpha = k/\rho c$. Initially the cylinder is at an ambient temperature of 70°F ; suddenly the inside wall temperature is raised to 1500°F and maintained at this temperature. All other conditions are the same as for the Steady State Problem.

An approximate analytical solution, which is valid until the temperature wave propagates to the outer wall of the cylinder, can be found by applying methods of solution discussed in Ref. 1. The result is:

$$T = (T_o - T_\infty) \sqrt{r_i/r} \operatorname{erfc} \left(\frac{r-r_i}{2\sqrt{\alpha t}} \right) + T_\infty$$

The finite element models employed here are identical to the models employed for the Steady State case. Transient solutions obtained by the Trapezoidal and Time Shape Function methods and using the linear element model converged to the same results obtained in the Steady State Problem for corresponding model. Solutions with Quadratic or Cubic models and Euler I integration were not pursued to steady state.

Tables VI through IX compare analytical and finite element solutions at various problem times for the 100 node, 24 Linear element model used with the Steady State Problem. Comparable results were obtained with Quadratic and Cubic element models. The following abbreviations apply to the time integration methods:

- a. E-1 Euler I
- b. TRAP Trapezoidal
- c. LTS Linear Time Shape Function

TABLE VI

RADIAL TEMPERATURE DISTRIBUTION IN A CYLINDER AT .05 MINUTES

Method	Temperature (°F)							
	Exact	E-1	E-1	TRAP	LTS	TRAP	LTS	
h(min)		.001	.0005	.001	.001	.01	.01	
Radius (in)								
3.00	1500.00	1500.00	1500.00	1500.00	1500.00	1500.00	1500.00	
3.25	492.38	595.41	591.70	593.06	592.26	636.45	629.17	
3.50	124.58	153.61	151.81	153.45	153.40	183.64	182.41	
3.75	72.81	63.84	63.99	64.64	64.91	69.94	72.42	
4.00	70.06	68.16	68.28	70.42	70.40	67.36	68.01	
5.00	70.00	70.01	70.01	70.00	70.00	70.00	70.00	
6.00	70.00	70.00	70.00	70.00	70.00	70.00	70.00	
9.00	70.00	70.00	70.00	70.00	70.00	70.00	70.00	

TABLE VII

RADIAL TEMPERATURE DISTRIBUTION IN A CYLINDER AT .50 MINUTES

Method	Temperature (°F)					
	Exact	E-1	E-1	TRAP	LTS	TRAP LTS
h (min)		.001	.0005	.001	.001	.01 .01
Radius (in)						
3.00	1500.00	1500.00	1500.00	1500.00	1500.00	1500.00
3.25	1096.15	1105.20	1104.93	1104.98	1104.91	1107.83 1107.10
3.50	756.59	770.04	769.60	769.70	769.59	774.45 773.35
3.75	495.82	509.04	508.58	508.73	508.62	514.12 513.10
4.00	313.60	323.52	323.15	323.33	323.27	328.28 327.64
5.00	80.88	80.34	80.34	80.42	80.45	81.20 81.46
6.00	70.11	70.03	70.03	70.03	70.03	70.04 70.06
7.00	70.00	70.00	70.00	70.00	70.00	70.00 70.00
9.00	70.00	70.00	70.00	70.00	70.00	70.00 70.00

TABLE VIII

RADIAL TEMPERATURE DISTRIBUTION IN A CYLINDER AT 1.00 MINUTES

Method	Temperature ($^{\circ}$ F)				
	Exact	E-1	E-1	TRAP	LTS
h (min)	.001	.0005	.001	.001	.01
Radius (in)					
3.00	1500.00	1500.00	1500.00	1500.00	1500.00
3.25	1195.88	1201.16	1201.08	1201.05	1201.87
3.50	928.00	936.13	935.98	935.93	937.41
3.75	701.28	710.30	710.13	710.08	712.02
4.00	517.45	525.93	525.78	525.72	527.90
5.00	145.20	147.10	147.11	147.11	148.54
6.00	76.24	76.00	76.03	76.04	76.37
7.00	70.24	70.18	70.18	70.18	70.21
8.00	70.00	70.00	70.00	70.00	70.00
9.00	70.00	70.00	70.00	70.00	70.00

TABLE IX
RADIAL TEMPERATURE DISTRIBUTION IN A
CYLINDER AT 1.50 MINUTES

Method	Temperature (°F)					
	Exact	E-1	TRAP	LTS	TRAP	LTS
h(min)		.0005*	.001**	.001**	.01	.01
Radius (in)						
3.00	1500.00	1500.00	1500.00	1500.00	1500.00	1500.00
3.25	1240.81	1245.47	1246.07	1246.00	1246.00	1245.84
3.50	1009.18	1016.34	1017.44	1017.33	1017.32	1017.04
3.75	806.91	815.06	816.53	816.38	816.37	816.01
4.00	634.79	642.85	644.53	644.38	644.35	643.98
5.00	220.69	224.17	225.54	225.50	225.39	225.29
6.00	95.63	95.97	96.47	96.50	96.42	96.50
7.00	72.69	72.57	72.66	72.68	72.65	72.70
8.00	70.17	70.14	70.15	70.15	70.15	70.15
9.00	70.01	70.01	70.01	70.01	70.01	70.01

*.0005 to 1 min, .001 thereafter

** .001 to 1 min., .01 thereafter

2. Transient Heat Conduction in a Prolate Spheroidal Solid

An analytical solution for transient heat conduction in a prolate spheroidal solid is developed in Ref. 4. Results of a finite element solution to this problem are presented in Refs. 10 and 11.

A non-dimensional finite element model using one cubic element or four Quadratic elements was constructed. The Linear Time Shape Function of time integration was employed giving results comparable to the results of Refs. 10 and 11. A comparison of theoretical and finite element results is presented in Figure 13.

D. SECOND ORDER TIME PROBLEMS

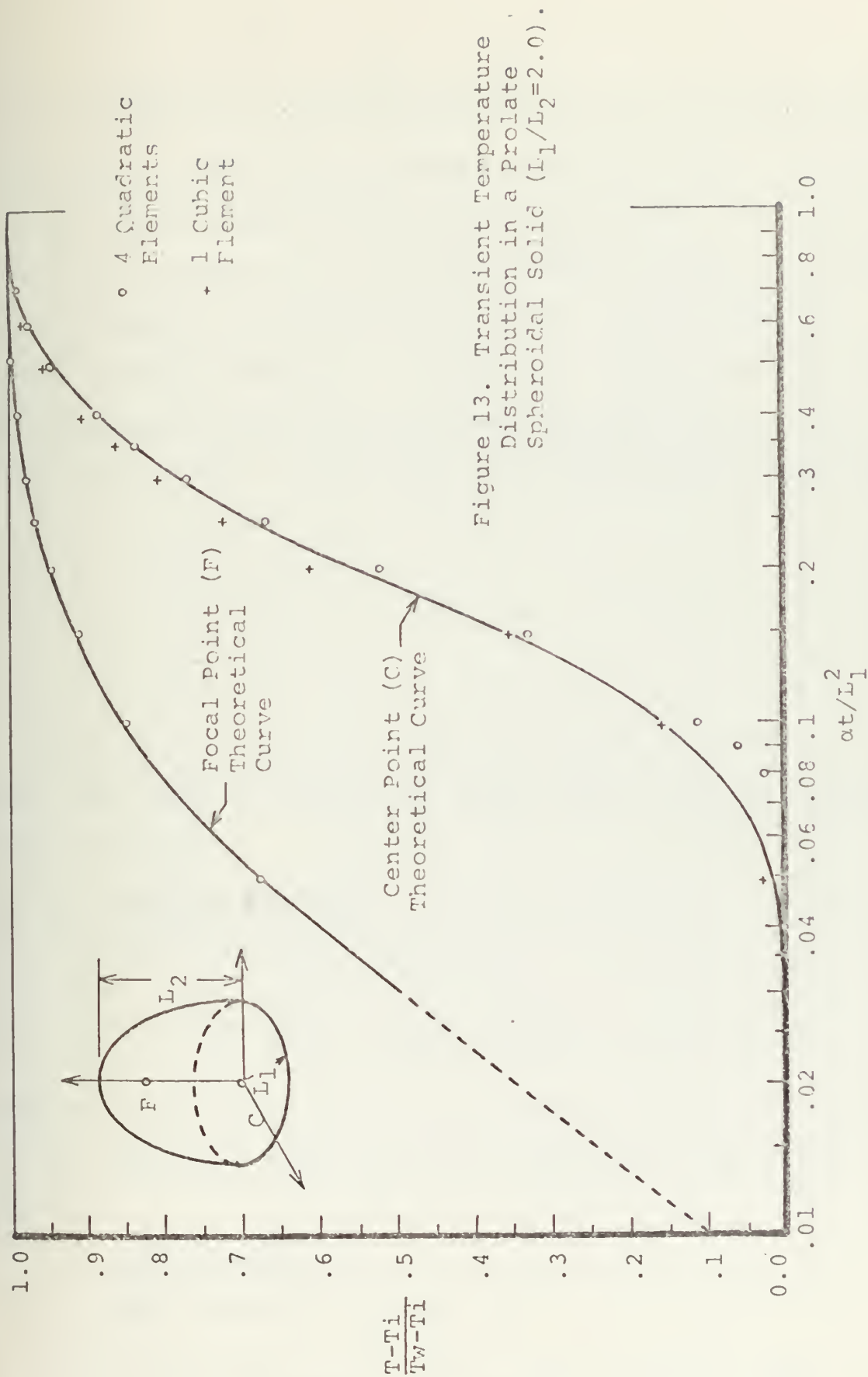
The transient vibration of a perfectly elastic membrane is a two-dimensional problem that can be formulated as a finite element model of the field equation.

Consider the square elastic membrane of length a and specified material properties T_0 , ρ , and v . T_0 is the initial tension of the membrane, ρ is the surface density, v is the damping factor, and $c^2 = T_0/\rho$. The partial differential equation describing the motion is:

$$c^2 \frac{\partial^2 u}{\partial x^2} + c^2 \frac{\partial^2 u}{\partial y^2} - 2v \frac{\partial u}{\partial t} - \frac{\partial^2 u}{\partial t^2} = 0$$

where $u(x,y,t)$ is the displacement of the membrane.

An exact mathematical solution is derived in Ref. 2 and may be specialized to a simple closed form if an initial displacement of the form



$$u(x,y,0) = d \sin \frac{\pi x}{a} \sin \frac{\pi y}{a}$$

and boundary conditions of simply supported edges are assumed.

Two finite element models were constructed. One models the entire membrane as a 36 Linear element plate; the other as a 9 Quadratic element quarter plate imposing symmetry boundary conditions at the centerlines. Euler I and Finite Difference solution techniques were employed with a time step size of .001. Assume the following material properties:

- a. $a = 1.5$
- b. $c^2 = 4.5$
- c. $v = 3.0$
- d. $d = .25$

Both time integration methods produced good results; however, the Euler I integration method was unstable at large time increments.

1. Undamped Membrane

The analytical solution is

$$u = d \cos \frac{2\pi ct}{a} \sin \frac{\pi x}{a} \sin \frac{\pi y}{a}$$

Results obtained with the finite element models are compared to the analytical result for the center node displacement in Table X.

2. Damped Membrane

The analytical solution is

$$u = d e^{-vt} (\cos \omega t + \frac{v}{\omega} \sin \omega t) \sin \frac{\pi x}{a} \cos \frac{\pi y}{a}$$

TABLE X
CENTER NODE DISPLACEMENT OF AN UNDAMPED MEMBRANE

Time	Displacement				
	EULER I Method			FINITE DIFFERENCE Method	
	Exact	Linear Model	Quadratic Model	Linear Model	Quadratic Model
.00	.2500	.2500	.2500	.2500	.2500
.05	.2378	.2372	.2370	.2370	.2368
.10	.2023	.2007	.2108	.2002	.2014
.15	.1469	.1439	.1456	.1435	.1453
.20	.0773	.0726	.0763	.0723	.0759
.25	.0000	-.0061	-.0020	-.0061	-.0018
.30	-.0773	-.0844	-.0787	-.0839	-.0784
.35	-.1469	-.1543	-.1496	-.1532	-.1483
.40	-.2023	-.2090	-.2041	-.2073	-.2027
.45	-.2378	-.2428	-.2407	-.2406	-.2383
.50	-.2500	-.2523	-.2516	-.2498	-.2495

where

$$\omega = \sqrt{\lambda^2 c^2 - v^2} \quad \text{and} \quad \lambda^2 = \pi/a$$

Results obtained with the finite element models are compared to the analytical result for the center node displacement in Table XI.

TABLE XI
CENTER NODE DISPLACEMENT OF A DAMPED MEMBRANE

Time	Displacement				
	Exact	EULER I Method		FINITE DIFFERENCE Method	
		Linear Model	Quadratic Model	Linear Model	Quadratic Model
.00	.2500	.2500	.2500	.2500	.2500
.05	.2389	.2384	.2382	.2382	.2380
.10	.2105	.2091	.2100	.2089	.2097
.15	.1716	.1692	.1705	.1691	.1704
.20	.1283	.1249	.1275	.1250	.1274
.25	.0854	.0813	.0841	.0815	.0843
.30	.0464	.0418	.0454	.0422	.0457
.35	.0135	.0089	.0122	.0095	.0128
.40	-.0119	-.0162	-.0128	-.0156	-.0123
.45	-.0298	-.0335	-.0309	-.0328	-.0301
.50	-.0405	-.0434	-.0411	-.0427*	-.0406*
.55	-.0450		**	-.0588	-.0587
.60	-.0445			-.0639	-.0651
.65	-.0404			-.0621	-.0644
.70	-.0340			-.0555	-.0585
.75	-.0263			-.0457	-.0491

* Changed step size to .01

** Changed step size to .01 at time = .5, Solution became unstable

V. CONCLUSIONS AND RECOMMENDATIONS

The computer program presented in this thesis provides an accurate and reliable means for solving a variety of problems. The use of this program and efforts to increase its versatility are highly encouraged.

The Linear Time Shape Function and Four Point Backward Finite Difference (Houbolt) time integration methods provide the most consistent and reliable results. Investigation of other time integration schemes should be pursued.

A. RECOMMENDED MODIFICATIONS TO THE PROGRAM

A number of modifications to the present program are desirable in order to increase the versatility of the program. Most of these modifications are straightforward and are the result of the simplifying assumptions made during development of the program. The recommended modifications should present no major programming problems.

1. Anisotropic Materials

A facility for coordinate axis rotation will provide the program with the ability to accommodate problems in which principal material axes do not align. This may be accomplished by the addition of an appropriate subroutine to perform the axis rotation. This subroutine should be called from MERGE at the time of element formation. Minor changes to subroutine INPUT will be required.

2. Time Dependent Boundary Conditions

The ability to account for a time variation in the boundary value specified for $\{\phi\}$ is highly desirable. In principle the modifications required are straightforward. The required modifications will only affect the solution subroutines TIME11, TIME13, TIME21, and TIME23 if data input is accomplished by providing updated values of $\{\phi\}$ and $\{\dot{\phi}\}$, if required, in tabular form. The modifications will require attention to detail and a thorough understanding of the Irons technique as applied in the program. At present, boundary values of $\{\dot{\phi}\}$ are considered to be zero; this effect on the solution matrix corrections and the effect of time dependence of the $\{F\}$ vector must be carefully considered.

Time dependence of material properties implies the recalculation of problem matrices at each station in time and possible iteration. Considerable computational effort is required and efficient programming techniques are necessary.

3. Non Linear Material Properties and Boundary Conditions

A functional dependence of material properties and non linear boundary conditions such as the radiation boundary condition of the heat problem are real physical situations. The ability of the program to accommodate this situation is needed and feasible by various iteration schemes.

4. Problem Size

The program is presently designed for problems of up to 110 equations with a maximum band width of 64. Practical

physical problems will require much more computer storage space. Modifications to accommodate more equations in core require a careful adjustment of the appropriate common and dimension statements and attention to the equivalence statements in the time integration subroutines.

5. Specialized Data Input and Output

Hand preparation of input data is tedious and subject to numerous errors. For practical physical problems of even moderate size, automated data input is a necessity.

Each class of physical problem will require data output in a specific form. Considerable attention should be given to the development of various subroutines to tailor data output to a given class of problem.

B. RECOMMENDED FUTURE STUDIES

This thesis was concerned with the development of a computer program with the capability of solving a class of physical problems. This objective was accomplished and recommendations for improvements stated. Questions raised during the course of this development and beyond the scope of this work are:

1. Convergence criteria of the various time integration methods, particularly the Euler I technique.
2. Selection criteria for finite element type and, in particular, the type of element best suited to a given geometry or physical situation.
3. Adaptability and usefulness of other finite element types.

APPENDIX A

COMPUTER PROGRAM NOMENCLATURE

A complete listing and description of all variables used in the program is not practical. The items listed in this appendix are common to several areas of the program and will assist the reader in a study of the program. For convenience items are listed by variable type. The definition or function and dimension, if applicable, of each item is given.

A. INTEGER CONSTANTS

IGO	Element type
IPUN	Punched output request
IPROB	Problem type
IPRT	Number of time increment steps to skip between print out of results
KBS	Boundary condition surface
KBT	Boundary condition type
KC	Coordinate system of reference
KINT	Time integration type
MRH	Time step size index
NBAND	System band width
NEL	Number of elements
NELBC	Number of element boundary conditions imposed
NGP	Number of Gauss points
NMAT	Number of materials
NNBC	Number of nodal boundary values specified

NNODE	Number of nodes
NPEL	Number of nodes per element
NTSC	Number of time step size increments specified
NTP	Number of times $\{\phi\}$ is to be printed

B. REAL CONSTANTS

ABIGN	Arbitrarily large number
AKX	[H] matrix property in the x direction
AKY	[H] matrix property in the y direction
AKZ	[H] matrix property in the z direction
APZRO	Calculated value of zero
DT	Time step size
DTJ	Determinant of the Jacobian
FACT	[C] or [G] matrix material property
PHIREF	Reference value of ϕ
TIM	Initial problem time
WFACT	Products of Gauss weight factors
WX,WY,WZ	Gauss weight factors for specified direction
XI,ETA,ZETA	Local coordinates ξ , η , and ζ
X1,Y1,Z1 or XX,YY,ZZ	Global coordinates x,y, and z

C. VECTORS

COL(32)	Element level storage of the shape function
ENDT(5)	End time of the various time integration steps
DTIM(5)	Time increment for each time step size
FE(32)	Forcing vector at element level
F(110)	Problem forcing vector

IP(5) IPRT for each time integration step size
 KBCOND(48) KBT for each element
 KEL(48) Element number for application of boundary conditions
 KNODE(110) Node number for specified $\{\phi_B\}$
 KSURF(48) KBS for each element
 PHI(110) $\{\phi\}$ vector
 PHIDOT(110) $\{\dot{\phi}\}$ vector
 PHIDDT(110) $\{\ddot{\phi}\}$ vector
 PHIWRK(110) Working vector
 TIME(6) Time label for output

D. MATRICES

AJ(3,3) [J] matrix
 AJI(3,3) $[J]^{-1}$ matrix
 BGC(110,64) [C] matrix
 BGG(110,64) [G] matrix
 BGM(110,64) [C] or [H] matrix, depending on problem type
 COARD(110,3) Global coordinates
 CORD(32,3) Global coordinates for element calculations
 DCOL(32,3) Derivatives of the shape function with respect to local coordinates. Column I is $\partial/\partial\xi$; Column II is $\partial/\partial\eta$; Column III is $\partial/\partial\zeta$.
 ENN(32,32) Work matrix for accumulation of $[H]^e$, $[C]^e$, or $[G]^e$.
 ENW(32,32) Work matrix for element level calculations
 NCON(48,33) Connectivity matrix for each element, column 1 is material number
 PROP(10,5) Material properties
 OUT(110,6) Equivalenced matrix for storage of output data

GP (6,5)	Gauss points
WF (6,5)	Gauss weighting factors

APPENDIX B

INSTRUCTIONS FOR PROBLEM DECK PREPARATION

Although the procedure is straightforward, preparation of input data for the program requires meticulous attention to detail. Errors are easy to make and difficult to locate.

Input data preparation should, in general, follow the steps outlined below before attempting to punch data cards. The use of the standard Fortran Eighty Column Coding Sheet is highly recommended. A sample problem deck follows this appendix. Integer constants must be right adjusted in the appropriate field.

Preliminary preparation:

1. Formulate the problem in terms of equation (1) and boundary conditions in terms of equations (2) and (3). Identify the appropriate material properties. In the case of a one or two-dimensional problem, certain properties must be specified as 1.0; a specification of 0.0 will lead to a singular system of equations.

2. Divide the continuum into the desired number and type of elements. Observe the restrictions of 48 elements, 110 nodes, 64 bandwidth, 10 materials, and 48 element level boundary conditions.

3. Assign node numbers as desired, observing the bandwidth restriction.

4. List the element connectivity in accordance with the conventions of Figures 1, 2, or 3. View ξ , η , and ζ as x , y , and z respectively. Any other convention will lead to serious error.

5. List all nodal points and coordinates in any convenient order.

6. List the applicable boundary conditions and the appropriate codes and material properties.

7. If applicable, select the solution subroutine desired and list the required time integration parameters. For the Houbolt method the specification of ENDT must allow enough time to generate the starting values required for the next step in time.

Preliminary preparations are now complete; precise card punching instructions follow. Card format is given in parenthesis followed by specific instructions where necessary. Recall that blank columns are read as zero.

1. Title Card (10A8)

Eighty columns on one card for any desired title information. Column 1 of this card will not be printed; if a 1 is punched, the output starts on a new page.

2. Problem Parameters (10I5,10X,F20.0)

One card containing the following data:

- a. Cols 1-5 : Number of nodes (NNODE)
- b. Cols 6-10 : Number of elements (NEL)
- c. Cols 11-15 : Number of materials (NMAT)
- d. Cols 16-20 : Number of Gauss points (NGP)
Specify 2 through 6; default to 3

- e. Cols 21-25 : Element type
 - 1 = Linear element
 - 2 = Quadratic element
 - 3 = Cubic element
- f. Cols 26-30 : Number of element level boundary conditions (NELBC)
- g. Cols 31-35 : Number of nodal boundary values specified (NNBC)
- h. Cols 36-40 : Problem type (IPROB)
 - 1 = Steady state
 - 2 = First order in time
 - 3 = Second order in time
 - 4 = Second order in time without [C] matrix
- i. Cols 41-45: Coordinate system of reference (KC)
 - 1 = Rectangular coordinates
 - 2 = Cylindrical coordinates
 - 3 = Spherical coordinates
- j. Cols 46-50: Punched output requested (IPUN)
 - 0 = No punched output desired
 - 1 = Punched output desired
- k. Cols 61-80: Reference value of ϕ (PHIREF)

3. Material Properties; One card per material for the steady state problem or two cards per material for the transient problem.

- a. Card I (I10,3F20.0)
 - Cols 1-10: Material number
 - Cols 11-30: k_x
 - Cols 31-50: k_y

Cols 51-70: k_z

b. Card II (I10,2F20.0)

Cols 1-10: Material number

Cols 11-30: [G] matrix coefficient

Cols 31-50: [C] matrix coefficient

4. Node cards; One set of cards for the steady state problem or two sets of cards for the transient problem.

Each set has NNODE cards.

a. Set I (I10,3F20.0)

Cols 1-10: Node number

Cols 11-30: x coordinate or radius

Cols 31-50: y coordinate or angle in degrees
relative to the x axis (+CCW)

Cols 51-70: z coordinate or angle in degrees
relative to the z axis (+CW)

b. Set II (I10,2F20.0)

Cols 1-10: Node number

Cols 11-30: Initial value of ϕ_i

Cols 31-50: Initial value of $\dot{\phi}_i$

Leave blank if first order time problem.

5. Node boundary values; NNBC cards required but omit
if NNBC = 0 (I10,F20.0)

Cols 1-10: Node number

Cols 11-30: Specified value of ϕ_i

6. Connectivity data; NEL cards required for the Linear
or Quadratic element and NEL x 2 cards required for the Cubic
element (2I5,32I3)

a. Card I (2I5,23I3)

Cols 1-5: Element number

Cols 6-10: Material identification number

Cols 11-79: Connectivity for nodes 1 through 23

b. Card II (9I3): Use with Cubic element only

Cols 1-27: Connectivity for cubic element
nodes 24-32

7. Boundary condition data; NELBC cards required but
omit if NELBC = 0 (3I5,F20.0)

Cols 1-5: Element number

Cols 6-10: Boundary condition type (KBT)

1 = Volume integral of loading (Q)

2 = Surface integral of loading (q)

3 = Surface loss coefficient (α)

Cols 11-15: Type of integral and surface
applicable (KBS)

0 = Volume integral; Use with KBT=1 only

± 1 = Surface integral over $\xi = \pm 1$ face

± 2 = Surface integral over $\eta = \pm 1$ face

± 3 = Surface integral over $\zeta = \pm 1$ face

Note: KBS = 1, 2, or 3 must be signed plus (+)
or minus (-) to indicate the
applicable integration surface

Cols 16-35: Boundary condition property value

8. Transient problem data; NTSC + 1 cards required but
omit for the steady state problem.

a. Card I, one card (2I10,10X,F20.0)

Cols 1-10: Number of changes in time step
size (NTSC)

Cols 11-20: Time integration method (KINT)
Default to method 3

1 = Euler I method

2 = Trapezoidal rule method

3 = Linear Time Shape Function or
Backward Finite Difference method

Cols 31-50: Initial problem time

b. Set II, NTSC cards (I10,2F20.0)

Cols 1-10: Number of time intervals to skip
before printing output (IPRT)

Cols 11-30: Time interval (DT)

Cols 31-50: Total problem time to end of
interval (ENDT)

Data preparation is now complete. Cards should be arranged in order of ascending node numbers within each set or group and each set or group of cards stacked in the order prepared. Refer to the sample problem deck following this appendix.

Several problems may be stacked and submitted if desired. In order to terminate a given computer run add one last data card with the word "STOP" punched in columns 1-4. Standard JCL cards and deck structure is shown in Appendix C. Print out of the program is suppressed by the "//FORT. SYSPRINT DD DUMMY" card; if a program listing is desired remove this card. The JCL card requesting additional output should be removed unless voluminous output is expected. Good Luck!

SAMPLE PROBLEM DECK

[illegible]

1	STEADY	32	STATE	AXIAL	HEAT	FLOW	IN AN	INSULATED	CYLINDER	(FLUX/TOPI)	<BTU/HR/FT>
1	2	1	20	1	2	1	8	1	0	20.	
2	1	1	2	2	82842712474619	20	82842712474619	1	4.	4.	
3	1	2	1	1	52814988653234	3	696587334865926	4.	4.	4.	
4	3	0	0	0		4.		2.	4.	4.	
5	4	0	0	0		0.		0.	4.	4.	
6	5	2	0	0		0.		0.	4.	4.	
7	6	4	0	0		0.		0.	4.	4.	
8	7	3	696587334865926	1	52814988653234	1	52814988653234	4.	4.	4.	
9	8	2	82842712474619	2	82842712474619	4.		3.	3.	3.	
10	9	0	0	0		0.		0.	3.	3.	
11	1	0	0	0		0.		0.	3.	3.	
12	1	2	4	2	82842712474619	2	82842712474619	2.	2.	2.	
13	1	3	2	82842712474619	3	696587334865926	4.	2.	2.	2.	
14	1	4	1	52814988653234	4		2.	2.	2.	2.	
15	1	5	0	0		2.		2.	2.	2.	
16	1	6	0	0		0.		2.	2.	2.	
17	1	7	0	0		0.		2.	2.	2.	
18	1	8	0	0		0.		2.	2.	2.	
19	1	9	4	3	696587334865926	0.		2.	2.	2.	
20	2	0	3	696587334865926	1	52814988653234	4.	1.	1.	1.	
21	2	1	2	82842712474619	2	82842712474619	4.	1.	1.	1.	
22	2	2	0	0		0.		1.	1.	1.	
23	2	3	0	0		0.		1.	1.	1.	
24	2	4	4	2	82842712474619	0.		1.	1.	1.	
25	2	5	2	82842712474619	2	82842712474619	3	696587334865926	0.	0.	
26	2	6	1	52814988653234	3	696587334865926	4.	0.	0.	0.	
27	2	7	0	0		4.		0.	0.	0.	
28	2	8	0	0		2.		0.	0.	0.	
29	2	9	0	0		0.		0.	0.	0.	
30	3	0	0	0		0.		0.	0.	0.	
31	3	1	4	3	696587334865926	1	52814988653234	0.	0.	0.	
32	3	2	3	696587334865926	1	52814988653234	4.	0.	0.	0.	

1234567890¹1234567890²1234567890³1234567890⁴1234567890⁵1234567890⁶1234567890⁷1234567890⁸

25 -200.
 26 -200.
 27 -200.
 28 -200.
 29 -200.
 30 -200.
 31 -200.
 32 -200.

1 1 13 3
 2 1 14 2
 3 1 15 3
 4 1 16 3
 5 1 17 3
 6 1 18 3
 7 1 19 3
 8 1 20 3
 9 1 21 3
 10 1 22 3
 11 1 23 3
 12 1 24 3
 13 1 25 3
 14 1 26 3
 15 1 27 3
 16 1 28 3
 17 1 29 3
 18 1 30 3
 19 1 31 3
 20 1 32 3

1
 2
 1
 STOP

APPENDIX C

STANDARD DECK STRUCTURE

```
//LEWSTHSI JOB (2032,0773FP,NS12), 'LEW,GIRARD T.',TIME=(1,30)
// EXEC FORTHCLG,REGION.GO=285K
//FORT.SYSPRINT DD DUMMY
//FORT.SYSIN DD *
```

Insert program deck at this point. Omit "DUMMY" card if a program listing is desired. Omit "SPACE" card if a large number of output pages is not required

```
/*
//GO.FT06F001 DD SPACE=(CYL,(4,1))
//GO.FT10F001 DD UNIT=SYSDA,SPACE=(CYL,(1,1)),DISP=(NEW,DELETE)
//GO.FT11F001 DD UNIT=SYSDA,SPACE=(CYL,(1,1)),DISP=(NEW,DELETE)
//GO.SYSIN DD *
```

Insert data deck here

```
/*
```


COMPUTER LISTING

```

** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** 
PROGRAM "FIELD", A SOLUTION TO THE TRANSIENT FIELD EQUATION USING
RESTRICTIONS
1: TIME DEPENDENCE IN BOUNDARY CONTITIONS NOT ALLOWED
2: STATIONARY ELEMENTS AND BOUNDARIES ARE REQUIRED
3: IF ANISOTROPIC THE PRINCIPAL AXES MUST ALIGN

INPUT DATA REQUIRED

(1) ONE TITLE CARD (10A8)
(2) PROBLEM DATA, ONE CARD (10I5,10X,F20.0)
1: NUMBER OF NODES (MAX IS 110) <NNODE>
MAX BAND WIDTH=64 <NBAND>
2: NUMBER OF ELEMENTS (MAX IS 48) <NEL>
3: NUMBER OF MATERIALS (MAX IS 10) <NMAT>
4: NUMBER OF GAUSS POINTS (2 TO 6; DEFAULT TO 3) <NGP>
5: CODE TO INDICATE ELEMENT TYPE:
1 = LINEAR ELEMENT
2 = QUADRATIC ELEMENT
3 = CUBIC ELEMENT
6: NUMBER OF ELEMENT BOUNDARY CONDITIONS (48 MAX) <NELBC>
7: NUMBER OF ELEMENTS WITH BOUNDARY VALUES SPECIFIED <NNBC>
8: PROBLEM TYPE <IPROB>
1 = STEADY STATE
2 = FIRST ORDER IN TIME
3 = SECOND ORDER IN TIME
4 = SECOND ORDER IN TIME, NO FIRST ORDER DERIVATIVE
9: INPUT = RECTANGULAR COORDINATES
1 = CYLINDRICAL COORDINATES
3 = SPHERICAL COORDINATES
10: PUNCHED OUTPUT REQUESTED <IPUN>
A POSITIVE INTEGER WILL PRODUCE AN OUTPUT OF NNODE
CARDS IN FORMAT (I10,2C24,16). NODE NUMBER IS
FOLLOWED BY THE VALUE OF "PHI" AND "PHIDOT" (IF
APPLICABLE). OUTPUT IS PRODUCED AT THE END OF EACH
TIME INTERVAL CHANGE FOR THE TRANSIENT PROBLEM.
11: REFERENCE VALUE OF PHI <PHIREF> (COLS 61-80)
SET AT ZERO IF NOT SPECIFIED

```

```

MAN0010
MAN0020
MAN0030
MAN0040
MAN0050
MAN0060
MAN0070
MAN0080
MAN0090
MAN0100
MAN0110
MAN0120
MAN0130
MAN0140
MAN0150
MAN0160
MAN0170
MAN0180
MAN0190
MAN0200
MAN0210
MAN0220
MAN0230
MAN0240
MAN0250
MAN0260
MAN0270
MAN0280
MAN0290
MAN0300
MAN0310
MAN0320
MAN0330
MAN0340
MAN0350
MAN0360
MAN0370
MAN0380
MAN0390
MAN0400
MAN0410
MAN0420
MAN0430

```


MAN0440
MAN0450
MAN0460
MAN0470
MAN0480
MAN0490
MAN0500
MAN0510
MAN0520
MAN0530
MAN0540
MAN0550
MAN0560
MAN0570
MAN0580
MAN0590
MAN0600
MAN0610
MAN0620
MAN0630
MAN0640
MAN0650
MAN0660
MAN0670
MAN0680
MAN0690
MAN0700
MAN0710
MAN0720
MAN0730
MAN0740
MAN0750
MAN0760
MAN0770
MAN0780
MAN0790
MAN0800
MAN0810
MAN0820
MAN0830
MAN0840
MAN0850
MAN0860
MAN0870
MAN0880
MAN0890
MAN0900
MAN0910

```

(3) MATERIAL PROPERTIES; NMAT X TWO CARDS
    (DATA ASSUMED ZERO IF NOT SPECIFIED)

CARD I: MATERIAL NUMBER
COLS 11-10: KX
COLS 11-30: KY
COLS 31-50: KZ
COLS 51-70: KZ

CARD II: (NOT REQUIRED FOR THE STEADY STATE PROBLEM)
COLS 11-10: MATERIAL NUMBER
COLS 11-30: DENSITY ("G" <SECOND TIME DERIATIVE> MATRIX
            COEFFICIENT)
COLS 31-50: VISCOSITY ("C" <FIRST TIME DERIVATIVE> MATRIX
            COEFFICIENT)

(4) NODE CARDS: TWO SETS OF NNODE CARDS PER PROBLEM
    (DATA ASSUMED ZERO IF NOT SPECIFIED)

SET I:
COLS 11-10 NODE NUMBER
COLS 11-30 X COORDINATE OR RADIUS
COLS 31-50 Y COORDINATE OR ANGLE IN DEGREES RELATIVE TO
            X AXIS (+ CCW)
COLS 51-70 Z COORDINATE OR ANGLE IN DEGREES RELATIVE TO
            Z AXIS (+ CW)

SET II: (NOT REQUIRED FOR THE STEADY STATE PROBLEM)
COLS 11-10: NODE NUMBER
COLS 11-30: INITIAL VALUE OF PHI
COLS 31-50: INITIAL VALUE OF PHIDOT

(5) NODE BOUNDARY VALUES: NNBC CARDS PER PROBLEM
    INPUT IN ORDER OF ACENDING NODE NUMBERS
    (OMIT IF DATA NOT REQUIRED)
COLS 11-10 JOINT NUMBER
COLS 11-30 SPECIFIED NODAL VALUE

(6) CONNECTIVITY DATA, NEL CARDS PER PROBLEM (215,3213)
COLS 1-5 ELEMENT NUMBER
COLS 6-10 MATERIAL IDENTITY, CCW FROM UPPER RIGHT
COLS 11-107 CONNECTIVITY, CCW FROM UPPER RIGHT
            LOOKING INTO Y-Z PLANE FROM (+) X AXIS
            NODES 1-23 ON CARD I (215,2313)
            NODES 24-32 ON CARD II (913)

(7) BOUNDARY CONDITION DATA: NELBC CARDS PER PROBLEM (315,F20.0)
    (OMIT IF DATA NOT REQUIRED)
  
```

CC

MAN0920
MAN0930
MAN0940
MAN0950
MAN0960
MAN0970
MAN0980
MAN0990
MAN1000
MAN1010
MAN1020
MAN1030
MAN1040
MAN1050
MAN1060
MAN1070
MAN1080
MAN1090
MAN1100
MAN1110
MAN1120
MAN1130
MAN1140
MAN1150
MAN1160
MAN1170
MAN1180
MAN1190
MAN1200
MAN1210
MAN1220
MAN1230
MAN1240
MAN1250
MAN1260
MAN1270
MAN1280
MAN1290
MAN1300
MAN1310
MAN1320
MAN1330
MAN1340
MAN1350
MAN1360
MAN1370
MAN1380
MAN1390

```
1: ELEMENT NUMBER (COLS 1-5)
2: CODE TO INDICATE BOUNDARY CONDITION TYPE (KBT)
   1 = VOLUME INTEGRAL OF LOADING
   2 = SURFACE INTEGRAL OF LOADING
   3 = SURFACE LOSS COEFFICIENT
3: CODE TO INDICATE TYPE OF INTEGRAL REQUIRED AND
   SURFACE APPLICABLE (KBS)
   0 = VOLUME INTEGRAL (USE WITH KBT = 1 ONLY)
   1 = SURFACE INTEGRAL OVER  $\xi = +/- 1$  FACE
   2 = SURFACE INTEGRAL OVER  $\eta = +/- 1$  FACE
   3 = SURFACE INTEGRAL OVER  $\zeta = +/- 1$  FACE
   *** KBT = 1, 2 OR 3 MUST BE SIGNED TO INDICATE ***
   *** PLUS (+) OR MINUS (-) FACE ***
4: BOUNDARY CONDITION PROPERTY VALUE (COLS 16-35)

(8) TRANSIENT PROBLEM DATA, NSTC + 1 CARDS
    (OMIT IF NOT REQUIRED)

CARD 1: (2110,10X,F20.0)
COLS 1-10: NUMBER OF CHANGES IN TIME STEP SIZE <NTSC>
          (MAXIMUM OF FIVE INCREMENTS ALLOWED)
COLS 11-20: TIME INTEGRATION METHOD <KINT>
          (DEFAULT TO METHOD THREE)
          1 = EULER 1 (FIRST AND SECOND ORDER TIME PROBLEM)
          2 = TRAPEZOIDAL (FIRST ORDER TIME PROBLEM ONLY)
          3 = TIME SHAPE FUNCTION (FIRST ORDER TIME PROBLEM)
          3 = HOUBOLT (SECOND ORDER TIME PROBLEM) "ENDT" MUST
              (FOR HOUBOLT ORDER TIME TO GENERATE THE
              ALLOW SUFFICIENT TIME TO GENERATE THE
              STARTING VALUES FOR THE NEXT TIME STEP
              SIZE CHANGE)
COLS 31-50: INITIAL PROBLEM TIME
          (SET AT ZERO IF NOT SPECIFIED)

SET 11: NTSC CARDS PER PROBLEM (110,2F20.0)
COLS 1-10: NUMBER OF TIME INTERVALS TO SKIP BEFORE
          PRINTING <IPRT>
COLS 11-30: TIME INTERVAL
COLS 31-50: TOTAL PROBLEM TIME TO END OF INTERVAL

STACK PROBLEMS IF DESIRED. TO TERMINATE RUN ADD DATA CARD
WITH "STOP" PUNCHED IN COLS 1-4

***** CODED IN NOVEMBER 1972 BY GIRARD T. LEW, LCDR, USN*****
*****
IMPLICIT REAL*8(A-H,O-Z)
COMMON /MTRX/ CURD(32,3), ENN(32,32), DCOL(32,3), COL(32,3), AJ(3,3), AJI
```



```

1(3,3),ENW(32,32),FE(32),PROPB(48),PROP(10,5),BGM(110,64),COORD(110
2,3),F(110),PHI(110),PHIDOT(110),PHIWRK(110),PHIDDT(110),DTIM(5),EN
3DT(5),BGH(110,64),BGC(110,64)
COMMON /FELP/ DTJ,AKX,AKY,AKZ,FACT,PHIREF,APZRO,ABIGN,TIM
COMMON /INT/ IGO,NBAND,NNODE,NMAT,NEL,NPEL,NGP,NELBC,NNBC,IPROB,KC
1,NTSC,KINT,IPJN
COMMON /IMTRX/ NCON(48,33),KEL(48),KBCOND(48),KSURF(48),KNODE(110)
1,IP(5)
100 CALL INPUT
GO TO (200,300,400,400),IPROB
C
C
C STEADY STATE PROBLEM
C
200 CALL MERGE(1)
CALL STEADY
GO TO 100
C
C
C FIRST ORDER TIME DERIVATIVE PROBLEM
C
C
300 CALL MERGE(1)
CALL MERGE(2)
IF(KINT.EQ.1) CALL TIME11
IF(KINT.EQ.2.DR.KINT.EQ.3) CALL TIME13
GO TO 100
C
C
C SECOND ORDER TIME DERIVATIVE PROBLEM
C
C
400 CALL MERGE(1)
IF(IPROB.EQ.3) CALL MERGE(2)
CALL MERGE(3)
IF(KINT.EQ.1) CALL TIME21
IF(KINT.EQ.2.DR.KINT.EQ.3) CALL TIME23
GO TO 100
END
MAN1430
MAN1430
MAN1410
MAN1420
MAN1430
MAN1430
MAN1440
MAN1450
MAN1460
MAN1470
MAN1480
MAN1490
MAN1500
MAN1510
MAN1520
MAN1530
MAN1540
MAN1550
MAN1560
MAN1570
MAN1580
MAN1590
MAN1600
MAN1610
MAN1620
MAN1630
MAN1640
MAN1650
MAN1660
MAN1670
MAN1680
MAN1690
MAN1700
MAN1710
MAN1720
MAN1730

```


IPT0010
 IPT0020
 IPT0030
 IPT0040
 IPT0050
 IPT0060
 IPT0070
 IPT0080
 IPT0090
 IPT0100
 IPT0110
 IPT0120
 IPT0130
 IPT0140
 IPT0150
 IPT0160

```

SUBROUTINE INPUT
  IMPLICIT REAL*8(A-H,O-Z)
  COMMON /MTRX/ CORD(32,3), ENN(32,32), DCOL(32,3), COL(32), AJ(3,3), AJ1
1(3,3), ENW(32,32), FE(32), PROPB(48), PROP(10,5), BGM(110,64), COARD(110
2(3), F(110), PHI(110), PHIDOT(110), PHIWRK(110), PHIDDT(110), DTIM(5), EN
3DT(5), BGC(110,64), BGC(110,64)
  COMMON /FLP/ JTJ, AKX, AKY, AKZ, FACT, PHIREF, APZRO, ABIGN, TIM
  COMMON /INT/ IGO, NBAND, NNODE, NMAT, NEL, NPCL, NGP, NELBC, NNBC, IPROB, KC
1, NTSC, KINT, IPJN
  COMMON /IMTRX/ NCON(48,33), KEL(48), KBCOND(48), KSURF(48), KNODE(110)
1, IP(5)
  DIMENSION TITLE (10)
  DATA CHK/, STOP //
  READ (5,1000) TITLE
  FORMAT (10A8)
1000 IF (TITLE(1).EQ.CHK) STOP
  
```

C
 C
 C
 READ THE BASIC PROBLEM DATA (1015,10X,F20.0)

```

  WRITE (6,1000) TITLE
  READ(5,1100) VNODE,NEL,NMAT,NGP,IGO,NELBC,NNBC,IPROB,KC,IPUN,PHIRE
  IF
1100 FORMAT(1015,10X,F20.0)
  WRITE (6,1200) NNODE,NEL,NMAT,NGP,NELBC,NNBC,IPROB,PHIREF
1200 FORMAT(/,/,11X, ' NUMBER OF NODES = ',I5,/,11X, ' NUMBER OF ELEMENT
  1S = ',I5,/,11X, ' NUMBER OF MATERIALS = ',I5,/,11X, ' NUMBER OF GA
  2USS POINTS = ',I5,/,11X, ' NUMBER OF ELEMENT BOUNDARY CONDITIONS =
  3 ',I5,/,11X, ' NUMBER WITH BOUNDARY VALUE = ',I5,/,11X, '
  4 PROBLEM TYPE = ',I5,/,11X, ' PHI(REFERENCE) = ',IPG24.16)
  
```

IPT0170
 IPT0180
 IPT0190
 IPT0200
 IPT0210
 IPT0220
 IPT0230
 IPT0240
 IPT0250
 IPT0260

C
 C
 C
 COMPUTE, THE NUMBER OF NODES PER ELEMENT FROM ELEMENT TYPE

```

  NPCL=12*(IGO-1)+8
  WRITE (6,1300) IGO, NPCL
1300 FORMAT (/,11X, ' TYPE',I3, ' ELEMENTS',I5, ' POINTS PER ELEMENT')
  IF(IPUN.GE.1) WRITE(6,1310)
1310 FORMAT(/,11X, ' PUNCHED OUTPUT OF "PHI" AND "PHIDOT" REQUESTED')
  
```

IPT0270
 IPT0280
 IPT0290
 IPT0300
 IPT0310

C
 C
 C
 ZERO THE DATA STORAGE MATRICES

```

  NP1=NPCL+1
  DO 1320 I=1,NEL
  DO 1320 J=1,NP1
1320 NCON(I,J)=0
  DO 1340 I=1,NNODE
  F(I)=0.0D0
  
```

IPT0320
 IPT0330
 IPT0340
 IPT0350
 IPT0360
 IPT0370

IPT0380
 IPT0390
 IPT0400
 IPT0410
 IPT0420
 IPT0430
 IPT0440
 IPT0450
 IPT0460
 IPT0470
 IPT0480
 IPT0490
 IPT0500
 IPT0510

IPT0520
 IPT0530
 IPT0540
 IPT0550
 IPT0560
 IPT0570
 IPT0580
 IPT0590
 IPT0600
 IPT0610
 IPT0620
 IPT0630

IPT0640
 IPT0650
 IPT0660
 IPT0670
 IPT0680
 IPT0690
 IPT0700
 IPT0710
 IPT0720
 IPT0730
 IPT0740
 IPT0750
 IPT0760
 IPT0770

```

    PHI(I)=0.000
    PHIDOT(I)=0.000
    PHIWRK(I)=0.000
    DO 1340 J=1,3
      COARD(I,J)=0.000
    DO 1360 I=1,NMAT
      DO 1360 J=1,5
        PROP(I,J)=0.000
    IF(NELBC.LE.0) GO TO 1390
    DO 1380 I=1,NELBC
      PROP(I)=0.000
    KBCOND(I)=0
    KSURF(I)=0
    KEL(I)=0
  1380 READ THE MATERIAL PROPERTIES
C
C
C
  1390 DO 1400 I=1,NMAT
    READ(5,1500) ID,(PROP(ID,J),J=1,3)
    IF(IPROB.LE.1) GO TO 1400
    READ(5,1500) ID,PROP(ID,4),PROP(ID,5)
  1400 CONTINUE
  1500 FORMAT(110,3F20.0)
    WRITE(6,1600)
  1600 FORMAT(//,32X,' MATERIAL PROPERTIES',//,' MTL# ',7X,' KX',18X,'
    1KY',18X,' KZ',15X,' DENSITY',12X,' VISCOSITY',//)
    DO 1700 I=1,NMAT
      DO 1700 J=1,5
        PROP(I,J)=0.000
    IF(IPROB.LE.1) GO TO 1400
    READ(5,1500) ID,PROP(ID,4),PROP(ID,5)
  1800 FORMAT(15,1P5G20.12,/)
C
C
C
C
    READ THE JOINT COORDINATES
    IF REQUIRED, READ BOUNDARY CONDITION NODAL VALUES AND/OR THE
    INITIAL VALUES OF PHI AND PHIDOT
  1850 DO 1850 I=1,NNODE
    READ(5,1875) IJT,(COORD(IJT,J),J=1,3)
  1875 FORMAT(110,3F20.0)
    IF(IPROB.LE.1) GO TO 2000
  1900 DO 1900 I=1,NNODE
    READ(5,1950) IJT,PHI(IJT),PHIDOT(IJT)
  1950 FORMAT(110,2F20.0)
  2000 IF(NNBC.LE.0) GO TO 2075
  2020 DO 2020 I=1,NNBC
    READ(5,2050) KNODE(I),PHIWRK(KNODE(I))
  2050 FORMAT(110,F20.0)
    WRITE(6,2060)
  2060 FORMAT(//,21X,' BOUNDARY CONDITION DATA',//,' NODE',18X,'VALUE',
    1///)
  
```



```

DO 2065 I=1,NNBC
ID=KNODE(I)
2065 WRITE(6,2070) ID,PHIWRK(ID)
2070 FORMAT(16,5X,1PG24.16)
2075 WRITE(6,2100)
2100 FORMAT(//,,' NODE #',10X,'X COORDINATE',12X,'Y COORDINATE',12X,'Z
COORDINATE',12X,'PHI INITIAL',9X,'PHIDOT INITIAL',//)
DO 2200 I=1,NNODE
2200 WRITE(6,2300) I,(COORD(I,J),J=1,3),PHI(I),PHIDOT(I)
2300 FORMAT(15,5X,1P5G24.16)
C
C IF COORDINATES NOT GIVEN IN RECTANGULAR SYSTEM, CONVERT
C
C IF(KC.GT.1) CALL MARK(COORD,KC,NNODE)
C
C READ ELEMENT CONNECTIVITY (215,2313,/,9I3)
C
DO 2400 I=1,NEL
2400 READ(5,2500) IEL,(NCON(IEL,J),J=1,NP1)
2500 FORMAT(215,2313,/,9I3)
2600 WRITE(6,2600)
2600 FORMAT(//,21X,' CONNECTIVITY MATRIX',//,2X,' EL# MTL#',//)
DO 2700 I=1,NEL
2700 WRITE(6,2800) I,(NCON(I,J),J=1,NP1)
2800 FORMAT(/,215,5X,20I4,/,15X,12I4)
C
C COMPUTE THE HALF BAND WIDTH FROM CONNECTIVITY DATA
C
NBAND=0
DO 3000 I=1,NEL
DO 2900 J=2,NPEL
JK=J+1
DO 2900 K=JK,NP1
NBAND=MAX0(NBAND,IABS(NCON(I,J)-NCON(I,K)))
3000 CONTINUE
NBAND=NBAND+1
WRITE(6,3100) NBAND
3100 FORMAT(//,11X,' SYSTEM HALF BAND WIDTH = ',15,//)
C
C READ IN BOUNDARY CONDITION DATA (315,F20.0)
C
IF (NELBC.LE.0) GO TO 3700
DO 3200 I=1,NELBC
3200 READ(5,3300) KEL(I),KBCOND(I),KSURF(I),PROPB(I)
3300 FORMAT(315,F20.0)
WRITE(6,3400)
3400 FORMAT(21X,' BOUNDARY CONDITION DATA',//,,' EL # BC # SURF
PROPERTY VALUE ',//)

```

IPT0780
 IPT0790
 IPT0800
 IPT0810
 IPT0820
 IPT0830
 IPT0840
 IPT0850
 IPT0860
 IPT0870

IPT0880

IPT0890
 IPT0900
 IPT0910
 IPT0920
 IPT0930
 IPT0940
 IPT0950
 IPT0960

IPT0970
 IPT0980
 IPT0990
 IPT1000
 IPT1010
 IPT1020
 IPT1030
 IPT1040
 IPT1050
 IPT1060

IPT1070
 IPT1080
 IPT1090
 IPT1100
 IPT1110
 IPT1120
 IPT1130


```

C
C
C
DO 3500 I=1,NELBC
3500 WRITE(6,3600) KEL(I),KBCOND(I),KSURF(I),PROPB(I)
3600 FORMAT(' ',3I5,1PG24.16)
C
C      READ THE TRANSIENT PROBLEM DATA (I10,2F20.0)
C
3700 IF(IPROB.LE.1) GO TO 4000
3725 READ(5,3725) NTSC,KINT,TIM
3725 IF(NTSC.LE.1) NTSC=1
DO 3750 I=1,NTSC
3750 READ(5,3800) IP(I),DTIM(I),ENDT(I)
3800 IF(KINT.LT.1.OR.KINT.GT.3) KINT=3
WRITE(6,3850) KINT,TIM
3850 FORMAT('///,21X,' INTEGRATION TYPE NUMBER =' ,15,///,21X,' START PRO
1BLEM AT TIME',1PG20.8,///,21X,' TIME INTEGRATION PARAMETERS',//,
2 SEGEMENT',6X,'DELTA TIME',15X,'PRINT SKIP',//)
DO 3900 I=1,NTSC
3900 WRITE(6,3950) I,DTIM(I),ENDT(I),IP(I)
3950 FORMAT(I10,1P2G25.16,I10)
4000 RETURN
END

```

IPT11140
 IPT11150
 IPT11160

IPT11170
 IPT11180
 IPT11190
 IPT11200
 IPT11210
 IPT11220
 IPT11230
 IPT11240
 IPT11250
 IPT11260
 IPT11270
 IPT11280
 IPT11290
 IPT11300
 IPT11310
 IPT11320
 IPT11330

MRG0010
MRG0020
MRG0030
MRG0040
MRG0050
MRG0051
MRG0060
MRG0070
MRG0071
MRG0080
MRG0081

```

SUBROUTINE MERGE(IMTRX)
  IMPLICIT REAL*8 (A-H,O-Z)
  COMMON /MTRX/ CORD(32,3), ENN(32,32), DCOL(32,3), COL(32), AJ(3,3), AJI
1(3,3), ENW(32,32), FE(32), PROPB(48), PROP(10,5), BGM(110,64), COARD(110
2,3), F(110), PHI(110), PHIDOT(110), PHIWRK(110), PHIDDT(110), DTIM(5), EN
3DT(5), BGC(110,64), BGC(110,64)
  COMMON /FLP/ DTJ, AKX, AKY, AKZ, FACT, PHIREF, APZRO, ABIGN, TIM
  COMMON /INT/ IGO, NBAND, NNODE, NMAT, NEL, NPEL, NGP, NELBC, NNBC, IPROB, KC
1, NTSC, KINT, IPJN
  COMMON /IMTRX/ NCON(48,33), KEL(48), KBCOND(48), KSURF(48), KNODE(110)
1, IP(5)

```

C
C

ZERO THE MASTER MATRIX

MRG0090
MRG0100
MRG0110
MRG0120

```

DO 1000 I=1, NNODE
DO 1000 J=1, N3AND
1000 BGM(I,J)=0.0D0
DO 8000 I=1, NEL

```

C
C

ZERO THE ELEMENT WORKING MATRICES

MRG0130
MRG0140
MRG0150
MRG0160
MRG0170
MRG0180
MRG0190
MRG0200
MRG0210
MRG0220
MRG0230

```

DO 1100 II=1, NPEL
DO 1110 JJ=1, 3
1110 CORD(II, JJ)=0.0D0
DO 1100 JI=1, NPEL
1100 ENW(II, JI)=0.0D0
DO 1200 J=1, NPEL
  INDX=NCON(I, J+1)
DO 1200 K=1, 3
1200 CORD(J, K)=COARD(INDX, K)
  IMAT=NCON(I, 1)
GO TO (2000, 3000, 4000), IMTRX

```

C
C

FORMING THE ELEMENTAL "H" MATRIX

MRG0240
MRG0250
MRG0260
MRG0270
MRG0280

```

2000 AKX=PROP(IMAT, 1)
  AKY=PROP(IMAT, 2)
  AKZ=PROP(IMAT, 3)
  CALL CUBE(NPEL, NGP, 1)
GO TO 6000

```

C
C

FORMING THE ELEMENTAL "C" MATRIX

MRG0290
MRG0300

```

3000 FACT=PROP(IMAT, 5)
GO TO 4500

```

C


```

C      FORMING THE ELEMENTAL "G" MATRIX
C      4000 FACT=PROP(I,MAT,4)
C      4500 CALL CUBE(NPEL,NGP,2)
C      DO 5000 II=1,NPEL
C      DO 5000 JJ=1,NPEL
C      5000 ENW(II,JJ)=FACT*ENW(II,JJ)
C      ASSEMBLING THE MASTER MATRIX
C      6000 DO 7000 II=1,NPEL
C      IDX=NCON(I,II+1)
C      DO 7000 JJ=1,NPEL
C      IDY=NCON(I,JJ+1)-IDX+1
C      IF (IDY.LT.1) GO TO 7000
C      BGM(IDX,IDY)=BGM(IDX,IDY)+ENW(II,JJ)
C      CONTINUE
C      7000 CONTINUE
C      8000
C      SKIP "BCOND" IF THERE ARE NO ELEMENTS WITH BOUNDARY CONDITIONS
C      IF(NELBC.GT.0.AND.IMTRX.EQ.1) CALL BCOND
C      OUTPUT MATRIX
C      IF(IMTRX.EQ.1.AND.IPROB.NE.1) CALL STUFF(BGM,BGH,NNODE,NBAND)
C      IF(IMTRX.EQ.2.AND.IPROB.EQ.3) CALL STUFF(BGM,BGC,NNODE,NBAND)
C      RETURN
C      END

```


BCD00010
BCD00020
BCD00030
BCD00040
BCD00050
BCD00060
BCD00070
BCD00080
BCD00090

```

SUBROUTINE BCND
  IMPLICIT REAL*8(A-H,O-Z)
  COMMON /MTRX/ CORD(32,3), ENN(32,32), DCOL(32,3), COL(32), AJ(3,3), AJI
1(3,3), ENW(32,32), FE(32), PROPB(48), PROP(10,5), BGM(110,64), COARC(110
2(3), F(110), PHI(110), PHIDOT(110), PHIWRK(110), PHIDDT(110), DTM(5), EN
3DT(5), BGH(110,64), BGC(110,64)
  COMMON /FLP/ DIJ, AKX, AKY, AKZ, FACT, PHIREF, APZRO, ABIGN, TIM
  COMMON /INT/ IGO, NBAND, NNODE, NMAT, NEL, NPEN, NGP, NELBC, NNBC, IPROB, KC
1, NTSC, KINT, IPUN
  COMMON /IMTRX/ NCON(48,33), KEL(48), KBCOND(48), KSURF(48), KNODE(110)
1, IP(5)

```

CCC

FORM THE "F" VECTOR

DO 800 I=1, NELBC

CCC

ZERO THE ELEMENT WORKING MATRICES

```

DO 200 II=1, NPEN
  FE(II)=0.0DO
  DO 150 JJ=1, 3
    CORD(II, JJ)=0.0DO
  DO 200 KK=1, NPEN
    ENW(II, KK)=0.0DO

```

150
200

CCC

EXTRACT ELEMENT BOUNDARY CONDITION DATA

```

IEL=KEL(I)
KBT=KBCOND(I)
KBS=KSURF(I)
CONST=PROPB(I)

```

BCD0160
BCD0170
BCD0180
BCD0190

CCC

EXTRACT ELEMENT COORDINATES

```

DO 300 J=1, NPEN
  INDX=NCON(IEL, J+1)
  DO 300 K=1, 3
    CORD(J, K)=CORD(INDX, K)
  CALL CUBEB(NPEN, NGP, KBT, KBS)
  GO TO (400, 400, 600), KBT

```

BCD0200
BCD0210
BCD0220
BCD0230
BCD0240
BCD0250

CCC

ASSEMBLE TYPE 1 OR TYPE 2 BOUNDARY CONDITION

```

400 DO 500 L=1, NPEN
  ID=NCON(IEL, L+1)
500 F(ID)=F(ID)+CONST*FE(L)

```

BCD0260
BCD0270
BCD0280

BCD00290

BCD00300
BCD00310
BCD00320

BCD00330
BCD00340
BCD00350
BCD00360
BCD00370
BCD00380
BCD00390
BCD00400
BCD00410
BCD00420

```
C
C
C      GO TO 800
C      ASSEMBLE TYPE 3 BOUNDARY CONDITION
C
C      600 DO 650 II=1,NPEL
C          DO 650 JJ=1,NPEL
C          ENW(II,JJ)=CONST*ENW(II,JJ)
C
C      ASSEMBLE BOUNDARY MATRIX INTO MASTER "H" MATRIX
C
C      DO 700 KK=1,NPEL
C          IDX=NCON(IEL,KK+1)
C          DO 700 LL=1,NPEL
C              IDY=NCON(IEL,LL+1)-IDX+1
C              IF (IDY.LT.1) GO TO 700
C              BGM(IDX,IDY)=BGM(IDX,IDY)+ENW(KK,LL)
C          CONTINUE
C      CONTINUE
C      RETURN
C      END
C
C      700
C      800
```


SDY0010
SDY0020
SDY0030
SDY0040
SDY0050
SDY0060
SDY0070
SDY0080
SDY0090
SDY0100
SDY0110

```

SUBROUTINE STEADY
IMPLICIT REAL*8(A-H,O-Z)
COMMON /MTRX/ CORD(32,3), ENN(32,32), DCOL(32,3), COL(32), AJ(3,3), AJI
1(3,3), ENW(32,32), FE(32), PROPB(48), PROP(10,5), BGM(110,64), CGARD(110
2(3,3), F(110), PHI(110), PHIDOT(110), PHIDDT(110), DTIM(5), EN
3DT(5), BGC(110,64), BGC(110,64)
COMMON /FLP/ DIJ, AKX, AKY, AKZ, FACT, PHIREF, APZRO, ABIGN, TIM
COMMON /INT/ IGO, NBAND, NNODE, NMAT, NEL, NPCL, NGP, NELBC, NNBC, IPROB, KC
1, NTSC, KINT, IPJN
COMMON /IMTRX/ NCON(48,33), KEL(48), KBCOND(48), KSURF(48), KNODE(110)
1, IP(5)

```

CCCCCCCC

SOLVING THE STEADY STATE PROBLEM OF THE FORM (H)*<PHI>+<F>=0
USING SUBROUTINES "LDLT" & "SLV"

APPLY THE SPECIFIED NODAL VALUES, PLACE "F" ON RIGHT HAND SIDE
STORE "F" FOR REFERENCE

SDY0090
SDY0091
SDY0100
SDY0110

```

DO 100 I=1,NNODE
PHI(I)=PHIWRK(I)
PHIWRK(I)=F(I)
100 F(I)=-F(I)

```

100

CCCC

FIND THE APPROXIMATE VALUE OF ZERO

CALL BNSN(BGM,NNODE,APZRO,ABIGN)

WRITE(6,150) APZRO,ABIGN

150 FORMAT('O',10X,'THE APPROXIMATE VALUE OF ZERO = ',1PG24.16,/,/,23X,

SDY0130
SDY0140
SDY0150
SDY0160
SDY0170
SDY0180
SDY0190
SDY0200
SDY0210
SDY0220
SDY0230
SDY0240
SDY0250
SDY0260
SDY0270
SDY0280
SDY0290
SDY0300
SDY0310

```

1 ABIGN = ,G24.16,/)
IF(NNBC.LE.0) GO TO 450
NEND=NBAND-1
DO 400 I=1,NNBC
ID=KNODE(I)
NCHK=NNODE-ID
DO 300 J=1,NEND
IROW=ID-J
ICOL=J+1
INDX=ID+J
IF(IROW.LE.0) GO TO 200
F(IROW)=F(IROW)-BGM(IROW,ICOL)*PHI(ID)
IF(J.GT.NCHK) GO TO 300
F(INDX)=F(INDX)-BGM(ID,ICOL)*PHI(ID)
CONTINUE
F(ID)=F(ID)-BGM(ID,1)*PHI(ID)
BGM(ID,1)=ABIGN

```

150

200

300


```

400 CONTINUE
450 CALL LDLT(BGM, NNODE, NBAND, APZRO)
C CALL SLV(BGM, F, NNODE, NBAND)
C
C FORM THE "PHI" AND "F" VECTORS
C
IF(NNBC.LE.0) GO TO 600
DO 500 I=1, NN3C
ID=KNODE(I)
PHIWRK(ID)=PHIWRK(ID)+ABIGN*F(ID)
500 F(ID)=PHI(ID)
600 WRITE(6,800)
800 FORMAT(1,' ', NODE', 7X, 'X COORDINATE', 8X, 'Y COORDINATE', 8X, 'Z COORD
1100 INATE', 4X, 'PHI(STEADY STATE)', 6X, 'FORCING VECTOR', 71X, 'ADJ TO R
2EF, ///)
DO 900 I=1, NNODE
PHI(I)=F(I)+PHIREF
900 WRITE(6,1000) I, (COORD(I,J), J=1,3), PHI(I), PHIWRK(I)
1000 FORMAT(15,1X,1P3G20.5,2G20.10)
TOT=0.0D0
DO 1100 I=1, NNODE
TOT=TOT+PHIWRK(I)
1100 WRITE(6,1200) TOT
1200 FORMAT(//,11X,' SUM OF FORCING VECTOR ELEMENTS IS',1PG20.10,/)
IF(IPUN.GE.1) CALL PUNC(F, PHI, NNODE, IPR08)
RETURN
END

```

SDY0320
SDY0330
SDY0340

SDY0350
SDY0360
SDY0370
SDY0380
SDY0390
SDY0400
SDY0410
SDY0420
SDY0430
SDY0440
SDY0450
SDY0460
SDY0470
SDY0480
SDY0490
SDY0500
SDY0510
SDY0520
SDY0521
SDY0530
SDY0540

T110010

SUBROUTINE TIME11

THIS SUBROUTINE EMPLOYS THE EULER ONE METHOD

T110020
T110030
T110040
T110050
T110060
T110070
T110080
T110090
T110100
T110110
T110120
T110130

```

IMPLICIT REAL*8(A-H,O-Z)
COMMON /MTRX/ CORD(32,3), ENN(32,32), DCOL(32,3), COL(32), AJ(3,3), AJ1
1(3,3), ENW(32,32), FE(32), PROPB(48), PROP(10,5), BGM(110,64), COARD(110
2,3), F(110), PHI(110), PHIWRK(110), PHIDOT(110), PHIDOT(110), DTIM(5), EN
3DT(5), BGH(110,64)
COMMON /FLP/ DTJ, AKX, AKZ, FACT, PHIREF, APZRO, ABIGN, TIM
COMMON /INT/ IGO, NBAND, NNODE, NMAT, NEL, NPCL, NGP, NELBC, NNBC, IPROB, KC
1, NTSC, KINT, IPUN
COMMON /IMTRX/ NCON(48,33), KEL(48), KBCOND(48), KSURF(48), KNODE(110)
1, IP(5)
DIMENSION OUT(110,6), TIME(6)
EQUIVALENCE (ENW(1,1), OUT(1,1))

```

```

PREPARE THE "C" MATRIX FOR THE SOLUTION PROCESS
INSERT BOUNDARY VALUE INTO PHI IF REQUIRED

```

T110140
T110150
T110160
T110170
T110180
T110190
T110200
T110210
T110220
T110230
T110240
T110250
T110260
T110270
T110280
T110290

```

CALL BNSN(BGM, NNODE, APZRO, ABIGN)
WRITE(6,100) APZRO, ABIGN
100 FORMAT(//,11X,'THE APPROXIMATE VALUE OF ZERO =',1PG24.16,/,23X,'
1 ABIGN =',G24.16,/)
IF(NNBC.LE.0) GO TO 400
DO 300 I=1,NN3C
ID=KNODE(I)
PHI(ID)=PHIWRK(ID)
BGM(ID,1)=ABIGN
300 CONTINUE
400 CALL LDLT(BGM, NNODE, NBAND, APZRO)
WRITE(6,500)
500 FORMAT('1,40X,'DISTRIBUTION OF PHI(ADJ TO REF) VS TIME ',/,52X,'
1 EULER METHOD',/)
DO 600 I=1,NNODE
OUT(I,1)=PHI(I)+PHIREF
600 OUT(I,1)=PHI(I)+PHIREF

```

PREPARE DATA FOR TIME INTEGRATION PROCESS

T110300
T110310
T110320
T110330
T110340
T110350
T110360

```

TIME(1)=0.000+TIM
IDP=1
DO 1400 MRH=1,NTSC
JE=6
IXX=-1
DT=DTIM(MRH)
TI=(ENDT(MRH)-TIM)/DT

```



```

SUBROUTINE TIME13
THIS SUBROUTINE EMPLOYS THE TIME SHAPE FUNCTION (LINEAR) METHOD
(KINT = 3) OR TRAPEZOIDAL (FINITE DIFFERENCE) METHOD (KINT = 2)

IMPLICIT REAL*8(A-H,O-Z)
COMMON /MTRX/ CORD(32,3), ENN(32,32), DCOL(32,3), COL(32), AJ(3,3), AJI
1(3,3), ENW(32,32), FE(32), PROPB(48), PROP(10,5), BGM(110,64), COARD(110
2,3), F(110), PHI(110), PHIDDT(110), PHIWRK(110), PHIDDT(110), DTIM(5), EN
3DT(5), BGM(110,64), BGC(110,64)
COMMON /FLP/ DTJ, AKX, AKZ, FACT, PHIREF, APZRO, ABIGN, TIM
COMMON /INT/ IGO, NBAND, NNODE, NMAT, NEL, NPCL, NGP, NELBC, NNBC, IPROB, KC
1, NTSC, KINT, IPUN
COMMON /IMTRX/ NCON(48,33), KEL(48), KBCOND(48), KSURF(48), KNODE(110)
1, IP(5)
1, DIMENSION OUT(110,6), TIME(6)
EQUIVALENCE(ENW(1,1), OUT(1,1))

PREPARE DATA FOR THE TIME INTEGRATION PROCESS
SAVE THE "F" VECTOR IN "PHIDDT" FOR RECOVERY AND REFERENCE

DO 25 I=1,NNODE
25 PHIDDT(I)=F(I)
TIM(1)=0.000+TIM
IDP=1

LOOP TO VARY TIME STEP SIZES

IF(NTSC.EQ.1) GO TO 50
REWIND 10
REWIND 11 BGM
WRITE(10) BGM
50 DO 1900 MRH=1,NTSC
IF(MRH.EQ.1) GO TO 75
REWIND 10
REWIND 11 BGM
75 READ(10) BGM
READ(11) BGM
JE=6
IXX=-1
DT=DTIM(MRH)
TI=(ENDT(MRH)-TIM)/DT
IPRT=IP(MRH)
ITIM=TI+1.5
NTP=(ITIM-1)/IPRT

```


1130360

IF(MRH.EQ.1) NTP=NTP+1

COMBINE THE "H" AND "C" MATRICES FOR THE SOLUTION PROCESS
LEFT SIDE MATRIX IS "BGM"; RIGHT SIDE MATRIX IS "BGH"

1130370
1130380
1130390
1130400
1130410
1130420
1130430
1130440
1130450
1130460
1130470
1130480
1130490
1130500

IF(KINT.EQ.3) GO TO 125
DO 100 I=1,NNODE
DO 100 J=1,NBAND
CIJ=BGM(I,J)/DT
AHIJ=BGH(I,J)/2.0D0
BGM(I,J)=AHIJ+CIJ
BGH(I,J)=AHIJ-CIJ
GO TO 175
100 DO 150 I=1,NNODE
125 DO 150 J=1,NBAND
CIJ=BGM(I,J)/DT
AHIJ=BGH(I,J)/3.0D0
BGM(I,J)=2.0D0*AHIJ+CIJ
150 BGH(I,J)=AHIJ-CIJ

PREPARE THE "LHS" MATRIX FOR THE SOLUTION PROCESS
TO ACCOMMODATE A TIME DEPENDENT "F" VECTOR, THE CORRECT ELEMENTS
FROM "BGM" MUST BE SAVED AND THE CORRECTION TO "F" APPLIED
IN THE TIME ITERATION LOOP

CCCCC

1130510
1130520
1130530
1130540
1130550
1130560
1130570
1130580
1130590
1130600
1130610
1130620
1130630
1130640
1130650
1130660
1130670
1130680
1130690
1130700
1130710
1130720
1130730

175 CALL BNSN(BGM,NNODE,APZRO,ABIGN)
WRITE(6,200) APZRO,ABIGN
200 FORMAT(//,11X,'THE APPROXIMATE VALUE OF ZERO =',1PG24.16,/,23X,'
1 ABIGN=',G24.16,/) GO TO 600
IF(NBAND-1
NEND=NBAND-1
DO 500 I=1,NNBC
ID=KNODE(I)
PHI(ID)=PHIWRK(ID)
NCHK=NNODE-ID
DO 400 J=1,NEND
IROW=ID-J
ICOL=J+1
INDX=ID+J
IF(IROW.LE.0) GO TO 300
F(IROW)=F(IROW)+BGM(IROW,ICOL)*PHI(ID)
300 IF(IROW.GT.NCHK) GO TO 400
F(INDX)=F(INDX)+BGM(ID,ICOL)*PHI(ID)
400 CONTINUE
F(ID)=F(ID)+BGM(ID,1)*PHI(ID)
500 BGM(ID,1)=ABIGN
600 CALL LDLT(BGM,NNODE,NBAND,APZRO)
IF(MRH.GT.1) GO TO 900


```

700 WRITE(6,700)
700 FORMAT(1,40X,'DISTRIBUTION OF PHI(ADJ TO REF) VS TIME',//)
700 IF(KINT.EQ.2) GO TO 740
720 WRITE(6,720)
720 FORMAT(47X,'TIME SHAPE FUNCTION METHOD',//)
720 GO TO 780
740 WRITE(6,760)
740 FORMAT(51X,'TRAPEZOIDAL METHOD',//)
760 DO 800 I=1,NNODE
780 OUT(I,1)=PHI(I)+PHIREF
800 OUT(I,1)=PHI(I)+PHIREF

      SOLUTION ITERATION LOOP

900 DO 1800 IT=1,ITIM
900 CALL MULT(BGH,PHI,PHIDOT,NNODE,NBAND)
1000 DO 1000 I=1,NNODE
1000 PHI(I)=-PHIDOT(I)-F(I)
1000 CALL SLV(BGM,PHI,NNODE,NBAND)
1000 IF(NNBC.LE.0) GO TO 1200
1000 DO 1100 I=1,NNBC
1000 ID=KNODE(I)
1000 SAV=ABIGN*PHI(ID)
1000 F(ID)=F(ID)+SAV
1000 PHIDOT(ID)=PHIDOT(ID)+SAV
1000 PHI(ID)=PHIWRK(ID)
1000 IXX=IXX+1
1000 IF(IXX.NE.IPRT) GO TO 1800
1000 IXX=0
1000 IDP=IDP+1
1000 TIME(IDP)=DT*(IT-1)+TIM
1000 DO 1300 I=1,NNODE
1000 OUT(I,IDP)=PHI(I)+PHIREF
1300

      WHEN OUT IS FULL, OUTPUT THE COLLECTED DATA

C
C
C
1400 IF(NTP.LT.JE) JE=NTP
1400 IF(IDP.LT.JE) GO TO 1800
1400 WRITE(6,1400) DT,(TIME(J),J=1,JE)
1400 FORMAT(//,70X,'TIME OF PHI DISTRIBUTION',//,70X,'DELTA TIME =',F2
1400 10.5,//,41X,1P6G15.6)
1400 WRITE(6,1500)
1400 FORMAT(//,' NODE',2X,'X CORD',6X,'Y CORD',6X,'Z CORD',2X,6(9X,'PH
1400 11,3X),//)
1400 DO 1600 I=1,NNODE
1400 WRITE(6,1700) I,(COORD(I,J),J=1,3),(OUT(I,K),K=1,JE)
1700 FORMAT(15,1P3G12.4,6G15.7)
1700 NTP=NTP+JE
1700 IDP=0

```


T131160

T131170
T131180
T131190
T131200
T131210
T131220

```
1800 CONTINUE  
C      RECOVER THE TRUE "F" VECTOR  
C  
      DO 1850 I=1,NNODE  
1850   F(I)=PHIDDT(I)  
      IF(I*PUN*GE.1) CALL PUNC(PHI,PHIDOT,NNODE,I*PROB)  
1900   TIM=DT*(ITIM-1)+TIM  
      RETURN  
      END
```


T210010
T210020
T210030
T210040
T210050
T210060
T210070
T210080
T210090
T210100
T210110
T210120
T210130

SUBROUTINE TIME21
THIS SUBROUTINE EMPLOYS THE EULER I METHOD

```
IMPLICIT REAL*8(A-H,O-Z)
COMMON /MTRX/ CORD(32,3), ENN(32,32), DCOL(32,3), COL(32), AJ(3,3), AJI
1(3,3), ENW(32,32), FE(32), PROPB(48), PROP(10,5), BGM(110,64), CGARD(110
2(3), F(110), PHI(110), PHIDOT(110), PHIWRK(110), PHIDDT(110), DTIM(5), EN
3DT(5), BGH(110,64), BGC(110,64)
COMMON /FLP/ DTJ, AKX, AKY, AKZ, FACT, PHIREF, APZRO, ABIGN, TIM
COMMON /INT/ IGO, NBAND, NNODE, NMAT, NEL, NPCL, NGP, NELBC, NNBC, IPROB, KC
1, NTSC, KINT, IPJN
COMMON /IMTRX/ NCON(48,33), KEL(48), KBCOND(48), KSURF(48), KNODE(110)
1, IP(5)
DIMENSION OUT(110,6), TIME(3)
EQUIVALENCE(ENW(1,1), OUT(1,1))
```

PREPARE THE "G" MATRIX FOR THE SOLUTION PROCESS
INSERT BOUNDARY VALUES INTO PHI IF REQUIRED

```
CALL BNSN(BGM, NNODE, APZRO, ABIGN)
WRITE(6,100) APZRO, ABIGN
FORMAT(//,11X, 'THE APPROXIMATE VALUE OF ZERO =',1PG24.16,/,23X, '
100 1 ABIGN =', G24.16,/)
IF(NNBC.LE.0) GO TO 400
DO 300 I=1, NNBC
```

T210140
T210150
T210160
T210170
T210180
T210190
T210200
T210210
T210220
T210230
T210240
T210250
T210260
T210270
T210280
T210290
T210300
T210310
T210320
T210330
T210340
T210350
T210360
T210370
T210380
T210390

```
ID=KNODE(I)
PHI(ID)=PHIWRK(ID)
BGM(ID,1)=ABIGN
CONTINUE
```

```
CALL LDLT(BGM, NNODE, NBAND, APZRO)
WRITE(6,800)
FORMAT(1,1,40X, 'DISTRIBUTION OF PHI(ADJ TO REF) AND PHIDOT VS TIME
1,/,56X, 'EULER ONE METHOD',/)
DO 900 I=1, NNODE
OUT(I,1)=PHI(I)+PHIREF
OUT(I,2)=PHIDOT(I)
TIME(I)=0.000+TIM
IDP=1
```

```
DO 1900 MRH=1, NTSC
JE=6
IXX=-1
DT=DTIM(MRH)
TI=(ENDT(MRH)-TIM)/DT
IPRT=IP(MRH)
ITIM=TI+1.5
```



```

C
C
NTP=(ITIM-1)/IPRT*2
IF(MRH.EQ.1) NTP=NTP+2

SOLUTION ITERATION LOOP

DO 1800 IT=1,ITIM
CALL MULT(BGH,PHI,PHIDDT,NNODE,NBAND)
DO 1000 I=1,NNODE
PHIDDT(I)=-PHIDDT(I)-F(I)
IF(1.PROB.EQ.4) GO TO 1200
CALL MULT(BGC,PHIDDT,PHIWRK,NNODE,NBAND)
DO 1100 I=1,NNODE
PHIDDT(I)=PHIDDT(I)-PHIWRK(I)
CALL SLV(BGM,PHIDDT,NNODE,NBAND)
IF(NNBC.LE.0) GO TO 1250
DO 1225 I=1,NNBC
ID=KNODE(I)
F(ID)=F(ID)+ABIGN*PHIDDT(ID)
PHIDDT(ID)=0.000
DO 1300 J=1,NNODE
PHI(J)=PHI(J)+DT*PHIDDT(J)
PHIDDT(J)=PHIDDT(J)+DT*PHIDDT(J)
IXX=IXX+1
IF(IXX.NE.IPRI) GO TO 1800
IXX=0
IDP=IDP+1
TIME(IDP)=DT*(IT-1)+TIM
IPD=2*IDP
LL=IPD-1
DO 1400 I=1,NNODE
OUT(I,LL)=PHI(I)+PHIREF
OUT(I,IPD)=PHIDDT(I)
1400

WHEN "OUT" IS FULL, PRINT THE COLLECTED DATA

IF(NTP.LT.JE) JE=NTP
L=JE/2
IF(IDP.LT.L) GO TO 1800
WRITE(6,1500) DT,(TIME(J),J=1,L)
1500 FORMAT(//,70X,'TIME OF PHI AND PHIDOT DISTRIBUTION',//,70X,'DELTA
1 TIME =',F20.5,//,41X,3(8X,1PG14.6,8X))
1550 WRITE(6,1550)
1550 FORMAT(//,' NODE',2X,'X CORD',6X,'Y CORD',6X,'Z CORD',2X,3(8X,'PHI
1',10X,'PHIDOT',3X),//)
DO 1600 I=1,NNODE
1600 WRITE(6,1700) I,(COORD(I,J),J=1,3),(OUT(I,K),K=1,JE)
1700 FORMAT(15,1P3G12.4,6G15.7)
IDP=0

```

T210400
T210410

T210420
T210430
T210440
T210450
T210460
T210470
T210480
T210490
T210500
T210510
T210520
T210530
T210540
T210550
T210560
T210570
T210580
T210590
T210600
T210610
T210620
T210630
T210640
T210650
T210660
T210670
T210680

T210690
T210700
T210710
T210720
T210730
T210740
T210750
T210760
T210770
T210780
T210790
T210800
T210810

T210820
T210830
T210840
T210850
T210860
T210870

```
1800 NTP=NTP-JE  
      CONTINUE  
1900 IF (IPUN.GE.1) CALL PUNC(PHI,PHIDOT,NNODE,IPOB)  
      TIM=DT*(ITIM-1)+TIM  
      RETURN  
      END
```


T230010

SUBROUTINE TIME23

THIS SUBROUTINE EMPLOYS A FOUR POINT BACKWARD FINITE DIFFERENCE
FORMULAE. STARTING VALUES ARE OBTAINED BY A TAYLOR SERIES
EXPANSION

```

IMPLICIT REAL*8(A-H,O-Z)
COMMON /MTRX/ CORD(32,3), ENN(32,32), DCOL(32,3), COL(32), AJ(3,3), AJI
1(3,3), ENW(32,32), FE(32), PROPB(48), PROP(10,5), BGM(110,64), COARD(110
2(3), F(110), PHI(110), PHIDOT(110), PHIWRK(110), PHIDT(110), DTIM(5), EN
3DT(5), BGH(110,64), BGC(110,64)
COMMON /FLP/ DTJ, AKX, AKY, AKZ, FACT, PHIREF, APZRO, ABIGN, TIM
COMMON /INT/ IGO, NBAND, NNODE, NMAT, NEL, NPCL, NGP, NELBC, NNBC, IPROB, KC
1, NTSC, KINT, IPUN
COMMON /IMTRX/ NCON(48,33), KEL(48), KBCOND(48), KSURF(48), KNODE(110)
1, IP(5)
DIMENSION OUT(110,6), TIME(6), SAVE(110,3), FSAV(110), START(110,3), PH
1ISAV(110), PHID(110)
EQUIVALENCE (ENW(1,1), OUT(1,1)), (ENN(1,1), SAVE(1,1)), (ENN(331), FSA
1V(1)), (ENN(441), START(1,1)), (ENN(771), PHISAV(1)), (ENW(881), PHID(1
2))

```

OBTAIN THREE STARTING VALUES OF PHI FROM A FIVE TERM TAYLOR
SERIES EXPANSION

```

REWIND 10
WRITE (10) BGM
REWIND 10
CALL BNSN(BGM, NNODE, APZRO, ABIGN)
DT=DTIM(1)
IF (NNBC.LE.0) GO TO 200
DO 100 I=1, NN3C
ID=KNODE(I)
PHI(ID)=PHIWRK(ID)
100 BGM(ID,1)=ABIGN
200 CALL LCLT(BGM, NNODE, NBAND, APZRO)
DO 300 I=1, NNODE
START(I,1)=PHI(I)
300 FSAV(I)=PHIDOT(I)
DO 1700 I=1, 2
ITP1=ITAIL+1
DO 400 I=1, NNODE
START(I, ITP1)=START(I, ITAIL)+DT*FSAV(I)
PHI(I)=START(I, ITP1)
400 PHIDOT(I)=FSAV(I)
DO 1700 J=1, 5

```


[illegible]

T230770
T230780
T230790
T230800
T230810

T230850
T230860
T230870
T230880
T230890
T230900
T230910
T230920
T230930
T230940
T230950
T230960
T230970
T230980
T230990
T231000
T231010
T231020

TIME(1)=0.000+TIM
IDP=1
DTSAV=1.000
REWIND 10
WRITE(10) BGH

LOOP TO VARY TIME STEP SIZE

DO 5300 MRH=1,NTSC
DT=DTIM(MRH)
TI=(ENDT(MRH)-TIM)/DT
IPRT=IP(MRH)
NTP=TI/IPRT
IF(MRH.EQ.1) GO TO 2100
REWIND 10
READ(10) BGH
IXX=-1

IS=1
GO TO 2200

2100

IS=3
IXX=1
NTP=NTP+1
JE=6
ITIM=TI+1.5
TOP=DTSAV*DTSV
BOT=DT*DT

2200

COMBINE THE "H", "C", & "G" MATRICES FOR THE SOLUTION PROCESS
LEFT SIDE MATRIX IS "BGH": RIGHT SIDE MATRICES ARE "BGC" AND "BGM"
REMOVE THE DELTA TIME CONTRIBUTION FROM "BGC" AND "BGM" IF
NECESSARY

IF(IPRQB.EQ.3) GO TO 2400

DO 2300 I=1,NNODE

DO 2300 J=1,NBAND

BGM(I,J)=TOP*BGM(I,J)/BOT

BGH(I,J)=BGH(I,J)+2.000*BGM(I,J)

GO TO 2600

DO 2500 I=1,NNODE

DO 2500 J=1,NBAND

BGM(I,J)=TOP*BGM(I,J)/BOT

BGC(I,J)=DTSV*BGC(I,J)/DT

BGH(I,J)=BGH(I,J)+2.000*BGM(I,J)+3.000*BGC(I,J)/2.000

PREPARE THE "LHS" MATRIX AND "F" VECTOR FOR THE SOLUTION PROCESS

CALL BNSN(BGH,NNODE,APZRO,ABIGN)

WRITE(6,2900) APZRO,ABIGN

T231140
T231150

T231030
T231040
T231050
T231060
T231070
T231080
T231090
T231100
T231110
T231120
T231130


```

3900  SAVE(I,J)=START(I,J)
      IM3=1
      IM2=2
      IM1=3
C
C
C      TIME ITERATION LOOP
      DO 5100  IT=IS,ITIM
      DO 4000  I=1,NNODE
      PHIDDT(I)=-SAVE(I,IM3)+4.0D0*SAVE(I,IM2)-5.0D0*SAVE(I,IM1)
      IF(I=PROB.EQ.4) GO TO 4200
      DO 4100  I=1,NNODE
      PHISAV(I)=SAVE(I,IM2)/2.0D0-2.0D0*SAVE(I,IM1)
      CALL MULT(BGC,PHISAV,PHIDDT,NNODE,NBAND)
      CALL MULT(BGM,PHIDDT,PHI,NNODE,NBAND)
      DO 4300  I=1,NNODE
      PHI(I)=-PHI(I)-F(I)-PHIDDT(I)
      CALL SLV(BGH,PHI,NNODE,NBAND)
      IF(NNBC.LE.0) GO TO 4500
      DO 4400  I=1,NNBC
      ID=KNODE(I)
      COR=ABIGN*PHI(ID)
      F(ID)=F(ID)+COR
      FSAV(ID)=FSAV(ID)+COR
      PHI(ID)=PHIWRK(ID)
      CALL STUFFV(SAVE,PHI,NNODE,IM3)
      IXX=IXX+1
      IM3=IM3+1
      IM2=IM2+1
      IM1=IM1+1
      IF(IM3.GT.3) IM3=1
      IF(IM2.GT.3) IM2=1
      IF(IM1.GT.3) IM1=1
      IF(IT.EQ.IM3S) CALL STUFFV(START,PHI,NNODE,1)
      IF(IT.EQ.IM2S) CALL STUFFV(START,PHI,NNODE,2)
      IF(IXX.NE.IPRT) GO TO 5100
      IXX=0
      IDP=IDP+1
      TIME(IDP)=DT*(IT-1)+TIM
      DO 4700  I=1,NNODE
      OUT(I,IDP)=PHI(I)+PHIREF
      4700  OUT(I,IDP)=PHI(I)+PHIREF
C
C
C      IF "OUT" IS FULL OUTPUT THE COLLECTED DATA
      IF(NTP.LT.JE) JE=NTP
      IF(IDP.LT.JF) GO TO 5100
      WRITE(6,4800) DT,(TIME(J),J=1,JE)
      4800  FORMAT(///,70X,'TIME OF PHI DISTRIBUTION',/,70X,'DELTA TIME =',F2

```

```

T231590
T231600
T231610
T231620

T231630
T231640
T231650
T231660
T231670
T231680
T231690
T231700
T231710
T231720
T231730
T231740
T231750
T231760
T231770
T231780
T231790
T231800
T231810
T231820
T231830
T231840
T231850
T231860
T231870
T231890
T231900
T231910
T231920
T231930
T231940
T231950
T231960
T231970

T231980
T231990
T232000
T232010

```


10.5, //, 42X, 1P6G15.6)	T232020
WRITE (6, 4850)	T232030
4850 FORMAT (//, ' NODE', 2X, 'X CORD', 6X, 'Y CORD', 6X, 'Z CORD', 4X, 6(9X, 'PH	T232040
1 I', 3X), //)	T232050
DO 4900 I=1, NNODE	T232060
4900 WRITE(6, 5000) I, (COORD(I, J), J=1, 3), (OUT(I, K), K=1, JE)	T232070
5000 FORMAT(15, 1P3G12.4, 6G15.7)	T232080
NTP=NTP-JE	T232090
IDP=0	T232100
5100 CONTINUE	T232110
CALL STUFFV(START, PHI, NNODE, 3)	T232220
C	
C	
C	
RECOVER THE TRUE "F" VECTOR AND "PHIDOT"	
DO 5200 I=1, NNODE	T232230
PHID(I) = (START(I, 2) - 4.000 * START(I, 3) + 3.000 * PHI(I)) / DT / 2.000	T232240
5200 F(I) = FSAV(I)	T232250
DTSAV = DT	T232260
IF (IPUN .GE. 1) CALL PUNC(PHI, PHID, NNODE, IPROB)	T232270
5300 TIM = DT * (ITIM - 1) + TIM	T232280
RETURN	T232290
END	T232300

SUBROUTINE	CUBE(NPEL,NGP,I TYPE)
IMPLICIT	REAL*8 (A-H,O-Z)
COMMON	/MTRX/ CORD(32,3),ENN(32,32),DCOL(32,3),COL(32),AJ(3,3),AJI
1	(3,3),ENW(32,32),FE(32),PROPB(48),PROP(10,5),BGM(110,64),COARD(110
2	,3),F(110),PHI(110),PHIDOT(110),PHWRK(110),PHIDT(110),DTIM(5),EN
3	DT(5),BGH(110,64),BGC(110,64)
COMMON	/FLP/ DTJ,AKX,AKY,AKZ,FACT,PHIREF,APZRO,ABIGN,TIM
DIMENSION	GP(6,5),WF(6,5)
DATA GP/	.5773502691896300,-.5773502691896300,0.000,0.000,0.000,0.000,0.000,
10	.7745966692414800,C.O.D0,-.7745966692414800,0.000,0.000,0.000,0.000,.86
21	11363115940500,0.3399810435848600,-.3399810435848600,-.86113631159
34	0500,0.000,0.000,9061798459386600,.5384693101056800,0.000,-.5384
46	93101056800,-9061798459386600,0.000,.9324695142031500,.66120938
56	4662600,.2386191860831700,-.2386191860831700,-.6612093864662600,-.
6	.9324695142031500/
DATA WF/	1.000,1.000,0.000,0.000,0.000,0.000,.5555555555555600,.888
1	8888888888900,.55555555555600,0.000,0.000,0.000,0.000,.347854845137450
20	,.6521451548625500,.6521451548625500,.3478548451374500,0.000,0.000,
33	,.2369268850561900,.4786286704993700,.568888888888900,.478628670
44	993700,.2369268850561900,0.000,.1713244923791700,.360761573048140
50	,.4679139345726900,.4679139345726900,.3607615730481400,.171324492
6	3791700/

۷۷۷

CHECK FOR AN OVERSPECIFIED GAUSS POINT NUMBER (DEFAULT TO THREE)

IF (NGP.LE.0.DR. NGP.GT.6) NGP=3
IDGP=NGP-1

UUU

SELECT THE INTEGRATION POINTS AND CALCULATE THE WEIGHT FACTORS

```

DO 300 I=1,NGP
XXI=GP(I, IDGP)
WX=WF(I, IDGP)
DO 300 J=1,NGP
ETA=GP(J, IDGP)
WY=WF(J, IDGP)
DO 300 K=1,NGP
ZETA=GP(K, IDGP)
WZ=WF(K, IDGP)
WFACT=WXX*WY*WZ
CUB0240
CUB0250
CUB0260
CUB0270
CUB0280
CUB0290
CUB0300
CUB0310
CUB0320
CUB0330

```

uuu

FORM THE ELEMENT PRODUCT MATRICES

```

IF ( ITYPE.EQ.1 ) GO TO 100
CALL FORNC(XI,ETA,ZETA,NPEL)
GO TO 200

```


CUB0370
CUB0380
CUB0390
CUB0400
CUB0410
CUB0420
CUB0430
CUB0440
CUB0450
CUB0460

```

100 CALL FORMH(XI,ETA,ZETA,NPEL)
200 DO 250 L=1,NPEL
    DO 250 M=1,NPEL
250   ENW(L,M)=ENW(L,M)+WFACT*ENN(L,M)
300 CONTINUE
    DO 350 I=1,NPEL
    DO 350 J=1,NPEL
350   ENW(I,J)=5.D-1*(ENW(I,J)+ENW(J,I))
    ENW(J,I)=ENW(I,J)
    RETURN

```

CCC

THE FOLLOWING PERTAINS TO BOUNDARY CONDITION CALCULATIONS ONLY

CUB0470
CUB0480
CUB0490
CUB0500
CUB0510

```

ENTRY CUBEB(NPEL,NGP,KBT,KBS)
IF (NGP.LE.0.OR.NGP.GT.6) NGP=3
IDGP=NGP-1
KBSABS=IABS(KBS)
GO TO (400,600,600),KBT

```

CCC

VOLUME INTEGRAL BOUNDARY CONDITIONS

CUB0520
CUB0530
CUB0540
CUB0550
CUB0560
CUB0570
CUB0580
CUB0590
CUB0600
CUB0610
CUB0620
CUB0630
CUB0640
CUB0650
CUB0660

```

400 DO 500 I=1,NGP
    XI=GP(I,IDGP)
    WX=WF(I,IDGP)
    DO 500 J=1,NGP
    ETA=GP(J,IDGP)
    WY=WF(J,IDGP)
    DO 500 K=1,NGP
    ZETA=GP(K,IDGP)
    WZ=WF(K,IDGP)
    WFACT=WX*WY*WZ
    CALL FORMV(XI,ETA,ZETA,NPEL)
    DO 450 L=1,NPEL
450   FE(L)=FE(L)-WFACT*COL(L)
500 CONTINUE
    GO TO 2000

```

CCC

SURFACE INTEGRAL BOUNDARY CONDITION

CUB0670
CUB0680
CUB0690
CUB0700
CUB0710
CUB0720
CUB0730
CUB0740
CUB0750

```

600 ALF=0.00
    C3=DFLOAT(KBS/KBSABS)
    DO 1500 I=1,NGP
    C1=GP(I,IDGP)
    W1=WF(I,IDGP)
    DO 1500 J=1,NGP
    C2=GP(J,IDGP)
    W2=WF(J,IDGP)
    WFACT=W1*W2

```


CUB0760
 CUB0770
 CUB0780
 CUB0790
 CUB0800
 CUB0810
 CUB0820
 CUB0830
 CUB0840
 CUB0850
 CUB0860
 CUB0870
 CUB0880
 CUB0890
 CUB0900
 CUB0910
 CUB0920
 CUB0930
 CUB0940
 CUB0950
 CUB0960
 CUB0970
 CUB0980
 CUB0990
 CUB1000
 CUB1010
 CUB1020
 CUB1030
 CUB1040

```

700 GO TO (700,800,900),KBSABS
    XI=C3
    ETA=C1
    ZETA=C2
800 GO TO 1000
    XI=C1
    ETA=C3
    ZETA=C2
900 GO TO 1000
    XI=C1
    ETA=C2
    ZETA=C3
1000 IF (KBT.EQ.3) GO TO 1200
    CALL FORMSQ(XI,ETA,ZETA,NPEL,KBSABS)
    DO 1100 L=1,NPEL
1100 FE(L)=FE(L)+WFACT*COL(L)
    GO TO 1500
1200 CALL FORMSA(XI,ETA,ZETA,NPEL,KBSABS)
    DO 1300 L=1,NPEL
    DO 1300 M=1,NPEL
1300 ENW(L,M)=ENW(L,M)+WFACT*ENN(L,M)
1500 CONTINUE
    IF (KBT.NE.3) GO TO 2000
    DO 1600 I=1,NPEL
    DO 1600 J=1,NPEL
1600 ENW(I,J)=5.0-I*(ENW(I,J)+ENW(J,I))
2000 RETURN
    END
  
```


SHP0390
 SHP0400
 SHP0410
 SHP0420
 SHP0430
 SHP0440
 SHP0450
 SHP0460
 SHP0470

```

ICOL=ICOL+1
XI=CORDX(IX)
YI=CORDY(IY)
ZI=CORDZ(IZ)
COL(ICOL)=FL(XI,ETA,ZETA,XI,YI,ZI)
DCOL(ICOL,1)=DFL(ETA,ZETA,XI,YI,ZI)
DCOL(ICOL,2)=DFL(ZETA,XI,YI,ZI,XI)
DCOL(ICOL,3)=DFL(XI,ETA,ZI,XI,YI)
GO TO 40
100
  
```

CCCCC

QUADRATIC ELEMENT FUNCTIONS

CORNER NODES (1,3,5,7,13,15,17,19)

```

20 DO 200 IZ=1,2
  II=12*(IZ-1)+1
  IT=II+6
  IX=0
  DO 200 ICOL=II,IT,2
    IX=IX+1
    YI=CORDY(IX)
    XI=CORDX(IY)
    ZI=CORDZ(IZ)
    COL(ICOL)=FQC(XI,ETA,ZETA,XI,YI,ZI)
    DCOL(ICOL,1)=DFQC(XI,ETA,ZETA,XI,YI,ZI)
    DCOL(ICOL,2)=DFQC(ETA,ZETA,XI,YI,ZI,XI)
    DCOL(ICOL,3)=DFQC(ZETA,XI,ETA,ZI,XI,YI)
200
  
```

MID SIDE NODES: 2, 6, 14, 18
4, 8, 16, 20
9, 10, 11, 12

CCCCC

```

IC9=8
DO 250 IZ=1,2
  II=12*(IZ-1)+2
  IT=II+4
  DO 250 IIC2=II,IT,4
    IC4=IC2+2
    IC9=IC9+1
    IN=IC9-8
    XI=CORDX(IN)
    YI=CORDY(IN)
    ZI=CORDZ(IN)
    COL(IC2)=FQM(XI,ETA,ZETA,ZI,YI)
    COL(IC4)=FQM(ETA,ZETA,XI,YI,-ZI)
    COL(IC9)=FQM(ZETA,XI,ETA,XI,YI)
    DCOL(IC2,1)=DFQM(XI,ETA,ZETA,ZI,YI)
  
```

SHP0610
 SHP0620
 SHP0630
 SHP0640
 SHP0650
 SHP0660
 SHP0670
 SHP0680
 SHP0690
 SHP0700
 SHP0710
 SHP0720
 SHP0730
 SHP0740
 SHP0750

SHP0760
 SHP0770
 SHP0780
 SHP0790
 SHP0800
 SHP0810
 SHP0820
 SHP0830
 SHP0840

SHP0850
 SHP0860
 SHP0870
 SHP0880
 SHP0890
 SHP0900
 SHP0910
 SHP0920
 SHP0930
 SHP0940
 SHP0950
 SHP0960
 SHP0970

SHP0980
 SHP0990
 SHP1000
 SHP1010
 SHP1020
 SHP1030
 SHP1040
 SHP1050
 SHP1060
 SHP1070
 SHP1080
 SHP1090
 SHP1100
 SHP1110
 SHP1120

```

DCOL(IC2,2)=DFQMY(XI,ZETA,Z1,Y1)
DCOL(IC2,3)=DFQMY(XI,ETA,Y1,Z1)
DCOL(IC4,1)=DFQMY(ETA,ZETA,-Z1,Y1)
DCOL(IC4,2)=DFQMX(ETA,XI,ZETA,-Z1,Y1)
DCOL(IC4,3)=DFQMY(ETA,XI,-Z1)
DCOL(IC9,1)=DFQMY(ZETA,ETA,X1,Y1)
DCOL(IC9,2)=DFQMY(ZETA,XI,Y1,X1)
DCOL(IC9,3)=DFQMX(ZETA,XI,ETA,X1,Y1)
GO TO 40
250
  
```

CCCCC

CUBIC ELEMENT FUNCTIONS

CORNER NODES (1,4,7,10/21,24,27,30)

```

30 DO 300 IZ=1,2
  II=20*(IZ-1)+1
  IT=II+9
  IX=0
  DO 300 ICOL=II,IT,3
    IX=IX+1
    XI=CORDX(IX)
    YI=CORDY(IX)
    ZI=CORDZ(IZ)
    COL(ICOL)=FCC(XI,ETA,ZETA,X1,Y1,Z1)
    DCOL(ICOL,1)=DFCC(XI,ETA,ZETA,X1,Y1,Z1)
    DCOL(ICOL,2)=DFCC(ETA,ZETA,X1,Y1,Z1,X1)
    DCOL(ICOL,3)=DFCC(ZETA,XI,ETA,Z1,X1,Y1)
300
  
```

CCCCC

MID SIDE NODES: 2, 9,22,29 // 3, 8,23,28
 5,12,25,32 // 6,11,26,31
 13,14,15,16, // 17,18,19,20

```

I13=12
DO 350 IZ=1,2
  ID=9-2*IZ
  IX=0
  DO 350 II=1,2
    IS=20*II-19+IZ
    IT=20*II-13-IZ
    DO 350 ICOL=IS,IT,ID
      I13=I13+1
      I5=ICOL+3
      IX=IX+1
      XI=CORDX(IX)
      YI=CORDY(IX)
      ZI=CORDZ(IX)
      THIRD=TRD(IZ)
350
  
```



```

SUBROUTINE JACOB (NPOL,AJ,DCOL,CORD,AJI,DTJ)
THIS SUBROUTINE WILL FORM THE JACOBIAN AND COMPUTE ITS DETERMINANT
AND INVERSE.
      3 X 3 JACOBIAN MATRIX
      AJI= 3 X 3 INVERSE OF THE JACOBIAN
      DTJ=DETERMINANT OF THE JACOBIAN
      CORD= NPOL X 3 MATRIX OF GLOBAL COORDINATES
      DCOL= NPOL X 3 MATRIX OF PARTIAL DERIVATIVES WITH RESPECT
              TO LOCAL COORDINATES
      NPOL= NUMBER OF NODES PER ELEMENT

IMPLICIT REAL*8 (A-H,O-Z)
DIMENSION AJ(3,3),AJI(3,3),CORD(32,3),DCOL(32,3)
FORM THE JACOBIAN
      DO 500 I=1,3
      DO 500 J=1,3
      AJ(I,J)=0.000
      DO 500 K=1,NPOL
      AJ(I,J)=AJ(I,J)+DCOL(K,I)*CORD(K,J)
      COMPUTE THE DETERMINANT
      DTJ=AJ(1,1)*AJ(2,2)*AJ(3,3)+AJ(1,2)*AJ(2,3)*AJ(3,1)+AJ(1,3)*AJ(2,1)*AJ(3,2)-
      1)*AJ(3,2)*AJ(3,1)*AJ(2,2)*AJ(1,3)-AJ(3,2)*AJ(2,3)*AJ(1,1)-AJ(3,3)*AJ(2,1)*AJ(1,2)
      COMPUTE THE INVERSE
      AJI(1,1)=(AJ(2,2)*AJ(3,3)-AJ(3,2)*AJ(2,3))/DTJ
      AJI(2,1)=-(AJ(2,1)*AJ(3,3)-AJ(3,1)*AJ(2,3))/DTJ
      AJI(3,1)=(AJ(2,1)*AJ(3,2)-AJ(3,1)*AJ(2,2))/DTJ
      AJI(1,2)=-(AJ(1,2)*AJ(3,3)-AJ(3,2)*AJ(1,3))/DTJ
      AJI(2,2)=(AJ(1,1)*AJ(3,3)-AJ(3,1)*AJ(1,3))/DTJ
      AJI(3,2)=-(AJ(1,1)*AJ(3,2)-AJ(3,1)*AJ(1,2))/DTJ
      AJI(1,3)=(AJ(1,2)*AJ(2,3)-AJ(2,2)*AJ(1,3))/DTJ
      AJI(2,3)=-(AJ(1,1)*AJ(2,3)-AJ(2,1)*AJ(1,3))/DTJ
      AJI(3,3)=(AJ(1,1)*AJ(2,2)-AJ(2,1)*AJ(1,2))/DTJ
      RETURN
THE FOLLOWING PERTAINS TO BOUNDARY CONDITION CALCULATIONS ONLY
ENTRY JACOB8(NPOL,AJ,DCOL,CORD,DTJ,KS)

```



```

C
C      FORM THE JACOBIAN
      DO 600 I=1,3
      DO 600 J=1,3
      AJ(I,J)=0.000
      DO 600 K=1,NPEL
        600 AJ(I,J)=AJ(I,J)+DCOL(K,I)*CORD(K,J)
C
C      COMPUTE THE SURFACE DETERMINANT
      GO TO (700,800,900),KS
      700 DTJ=AJ(2,2)*AJ(3,3)-AJ(2,3)*AJ(3,2)
      GO TO 1000
      800 DTJ=AJ(1,1)*AJ(3,3)-AJ(3,1)*AJ(1,3)
      GO TO 1000
      900 DTJ=AJ(1,1)*AJ(2,2)-AJ(2,1)*AJ(1,2)
      1000 RETURN
      END
C
C      JACO230
C      JACO240
C      JACO250
C      JACO260
C      JACO270
C
C      JACO280
C      JACO290
C      JACO300
C      JACO310
C      JACO320
C      JACO330
C      JACO340
C      JACO350

```



```

100 SUBROUTINE MARK(C,K,M)
    IMPLICIT REAL*8(A-H,O-Z)
    DIMENSION C(110,3)
    PI=3.141592653589793D0
    IF(K.NE.2) GO TO 300
    DO 100 I=1,M
        ANG=C(I,2)*PI/1.8D2
        R=C(I,1)*DCOS(ANG)
        C(I,2)=R*DSIN(ANG)
        WRITE(6,200)
200   FORMAT('O',5X,'COORDINATES GIVEN IN CYLINDRICAL SYSTEM OF REFERENCE',10
1E,'//,6X,'RECTANGULAR COORDINATES FOLLOW',//,5X,'NODE NUMBER',10
2X,'X COORDINATE',12X,'Y COORDINATE',12X,'Z COORDINATE',//)
        GO TO 600
300   DO 400 I=1,M
        ANGZ=C(I,3)*PI/1.8D2
        ANGZ=C(I,2)*PI/1.8D2
        R=C(I,1)
        SIN=DSIN(ANGZ)
        C(I,1)=R*SIN*DCOS(ANGX)
        C(I,2)=R*SIN*DSIN(ANGX)
        C(I,3)=R*DCOS(ANGZ)
        WRITE(6,500)
500   FORMAT('O',5X,'COORDINATES GIVEN IN SPHERICAL SYSTEM OF REFERENCE',
1E,'//,6X,'RECTANGULAR COORDINATES FOLLOW',//,5X,'NODE NUMBER',10X,
2X,'X COORDINATE',12X,'Y COORDINATE',12X,'Z COORDINATE',//)
        DO 700 J=1,M
700   WRITE(6,800) J,C(J,L),L=1,3)
800   FORMAT(10X,15,5X,1P3G24.16)
    RETURN
END

```

MAR0010
MAR0020
MAR0030
MAR0040
MAR0050
MAR0060
MAR0070
MAR0080
MAR0090
MAR0100
MAR0110
MAR0120
MAR0130
MAR0140
MAR0150
MAR0160
MAR0170
MAR0180
MAR0190
MAR0200
MAR0210
MAR0220
MAR0230
MAR0240
MAR0250
MAR0260
MAR0270
MAR0280
MAR0290
MAR0300
MAR0310
MAR0320


```

SUBROUTINE BNSN(A,N,APZRO,ABIGN)
IMPLICIT REAL*8(A-H,O-Z)
DIMENSION A(110,64)
ANB=0.0D0
DO 100 I=1,N
  ANB=ANB+DABS(A(I,1))
APZRO=ANB*1.0D-15/DFLOAT(N)
ABIGN=APZRO*1.0D30
RETURN
END
100

```

BSN0010
 BSN0020
 BSN0030
 BSN0040
 BSN0050
 BSN0060
 BSN0070
 BSN0080
 BSN0090
 BSN0100

```

SUBROUTINE MULT(A,B,C,N,M)
IMPLICIT REAL*8(A-H,O-Z)
DIMENSION A(110,64),B(110),C(110)
DO 200 I=1,N
  IST=I-M-1
  C(I)=B(I)*A(I,1)
  DO 200 J=2,M
    IROW=IST+J
    IF(IROW.LE.0) GO TO 100
    ICOL=M-J+2
    C(I)=C(I)+A(IROW,ICOL)*B(IROW)
  100 IR=I+J-1
    IF(IR.GT.N) GO TO 200
    C(I)=C(I)+A(I,J)*B(IR)
  CONTINUE
RETURN
END
100
200

```

MUT0010
 MUT0020
 MUT0030
 MUT0040
 MUT0050
 MUT0060
 MUT0070
 MUT0080
 MUT0090
 MUT0100
 MUT0110
 MUT0120
 MUT0130
 MUT0140
 MUT0150
 MUT0160
 MUT0170

STFOOL O

SUBROUTINE STJFF(A,B,N,M)

~~~~~

THIS SUBROUTINE WILL PLACE THE CONTENTS OF MATRIX "A" INTO MATRIX "B"; MATRIX "A" IS PRESERVED; N = NUMBER OF ROWS; M = NUMBER OF COLUMNS

STSTST  
TFTFTF  
FOFOFO  
OOCOOC  
2345678  
0000000

IMPLICIT REAL\*8(A-H,C-Z)  
DIMENSION A(110,64),B(110,64)

100

$$B(I, J) = A(I, J)$$

REFITURN

COUNTRY STUFFV(AA,BB,N,MC)

۷۷۷

PLACES VECTOR "BB" INTO SPECIFIED COLUMN (MC) OF MATRIX "AA"

STEF0120  
STEF0120  
STEF0120  
STEF0120  
STEF0120

DIMENSION AA(110,3),88(110)

DI MENS I UN' A A  
DO 200 I = 1. N

$$AA(I, MC) = 3B(I)$$

RETURN

Y W

PUN00000000  
PUN00000001  
PUN00000002  
PUN00000003  
PUN00000004  
PUN00000005  
PUN00000006  
PUN00000007  
PUN00000008  
PUN00000009  
PUN00000010  
PUN00000011  
PUN00000012  
PUN00000013

SUBROUTINE PUNC(A,B,N,K)

SUBROUTINE DZM(A,D,N,Z)  
IMPLICIT REAL\*8(A-H,O-Z)

IMPLICATION REALT<sup>8</sup>(A-H, D-Z)  
DIMENSION ALLIO). B(110)

IF (K.GT.2) GO TO 305

IF (K.GI.Z) DO  
DO 100 I=1,N

WITTENBERG, I. (1977, 2002) I. A. (I)

WRIE(7,200) I.A(11)  
FORMAT(110,D24,16)

FORM 110  
GO TO 600

GO 10 600  
DO 400 I=1..N

```
DO 400 I=1,N
WRITE(7,50.0) I,A(I),B(I)
```

WRITEL: 509) I: A(11)  
EUBMAT (I10: 2024-16)

FORTRAN

WZU





## LIST OF REFERENCES

1. Arpaci, V. S., Conduction Heat Transfer, Addison-Wesley Publishing Company, 1966.
2. Berg, P. W. and McGregor, J. L., Elementary Partial Differential Equations, Holden-Day, 1966.
3. Crandall, S. H., Engineering Analysis, McGraw-Hill, 1956.
4. Haji-Sheikh, A. and Sparrow, E. M., "Transient Heat Conduction in a Prolate Spheroidal Solid," Trans. ASME, v. 88, p. 331-3, August 1966.
5. Holman, J. P., Heat Transfer, 2d ed., Mc-Graw Hill, 1963.
6. Irons, B. M., "A Frontal Solution Program for Finite Element Analysis," International Journal for Numerical Methods in Engineering, v. 2, p. 5-32, 1970.
7. Johnson, D. E., "A Proof of the Stability of the Houbolt Method," AIAA Journal, v. 4, No. 7, p. 1450-1451, August 1966.
8. Newmark, N. M., "A Method of Computation for Structural Dynamics," ASCE, Journal of Engineering Mechanics Division, v. 85, No. EM3, p. 67-64, July 1959.
9. Stroud, A. H. and Secrest, D., Gaussian Quadrature Formulas, Prentice-Hall Inc., 1966.
10. Zienkiewicz, O. C., The Finite Element Method in Engineering Science, McGraw-Hill, 1971.
11. Zienkiewicz, O. C. and Parekh, C. J., "Transient Field Problems: Two-Dimensional and Three-Dimensional Analysis by Isoparametric Finite Elements," International Journal for Numerical Methods in Engineering, v. 2, No. 1, p. 61-71, January-March 1970.
12. Users Manual, W. R. Church Computer Center, U.S. Naval Postgraduate School, Monterey, California, 1970.
13. Private communication from Gilles Cantin.



# INITIAL DISTRIBUTION LIST

|                                                                                                                                               | No. Copies |
|-----------------------------------------------------------------------------------------------------------------------------------------------|------------|
| 1. Defense Documentation Center<br>Cameron Station<br>Alexandria, Virginia 22314                                                              | 2          |
| 2. Library, Code 0212<br>Naval Postgraduate School<br>Monterey, California 93940                                                              | 2          |
| 3. Professor Gilles Cantin, Code 59Ci<br>Department of Mechanical Engineering<br>Naval Postgraduate School<br>Monterey, California 93940      | 5          |
| 4. Asst Professor David Salinas, Code 59Zc<br>Department of Mechanical Engineering<br>Naval Postgraduate School<br>Monterey, California 93940 | 5          |
| 5. LCDR Girard T. Lew<br>Supervisor of Shipbuilding, Conversion,<br>and Repair, USN<br>Pascagoula, Mississippi 39567                          | 3          |
| 6. Professor Paul F. Pucci, Code 59Pc<br>Department of Mechanical Engineering<br>Naval Postgraduate School<br>Monterey, California 93940      | 1          |
| 7. Professor Robert E. Newton, Code 59Ne<br>Department of Mechanical Engineering<br>Naval Postgraduate School<br>Monterey, California 93940   | 1          |
| 8. Professor John E. Brock, Code 59Bc<br>Department of Mechanical Engineering<br>Naval Postgraduate School<br>Monterey, California 93940      | 1          |
| 9. Dr. Arnold Adicoff<br>Naval Weapons Center<br>China Lake, California 93555                                                                 | 5          |
| 10. W. R. Church Computer Center, 0211<br>Naval Postgraduate School<br>Monterey, California 93940                                             | 2          |



UNCLASSIFIED

Security Classification

## DOCUMENT CONTROL DATA - R &amp; D

Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

SPONSORING ACTIVITY (Corporate author)

Naval Postgraduate School  
Monterey, California 93940

2a. REPORT SECURITY CLASSIFICATION

Unclassified

2b. GROUP

TITLE

THREE DIMENSIONAL SOLUTION OF THE TRANSIENT FIELD PROBLEM  
USING ISOPARAMETRIC FINITE ELEMENTS

ADDITIONAL NOTES (Type of report and, inclusive dates)

Master's Thesis; December 1972

AUTHOR (First name, middle initial, last name)

Erard T. Lew; Lieutenant Commander, United States Navy

DATE

December 1972

7a. TOTAL NO. OF PAGES

122

7b. NO. OF REFS

13

TRACT OR GRANT NO.

9a. ORIGINATOR'S REPORT NUMBER(S)

JECT NO.

9b. OTHER REPORT NO(S) (Any other numbers that may be assigned  
this report)

DISTRIBUTION STATEMENT

Approved for public release; distribution unlimited.

ADDITIONAL NOTES

12. SPONSORING MILITARY ACTIVITY

Naval Postgraduate School  
Monterey, California 93940

ABSTRACT

An isoparametric finite element formulation solving a general form of the transient field equation is presented in this thesis. The formulation is developed for fields in three-dimensional Euclidean space. A FORTRAN IV computer program employing double precision arithmetic, compact storage techniques, and providing the option of several numerical integration methods and types of finite elements is presented. Data input may be in rectangular, cylindrical, or spherical coordinates and variations in time integration step size are permitted. Time dependent boundary conditions are not considered.

Comparisons of theoretical and computer solutions for a variety of test problems demonstrate close agreement. Instructions for use of the program are included.



| KEY WORDS                    | LINK A |    | LINK B |    | LINK C |    |
|------------------------------|--------|----|--------|----|--------|----|
|                              | ROLE   | WT | ROLE   | WT | ROLE   | WT |
| FINITE ELEMENT               |        |    |        |    |        |    |
| FIELD                        |        |    |        |    |        |    |
| TRANSIENT FIELD              |        |    |        |    |        |    |
| ISOPARAMETRIC FINITE ELEMENT |        |    |        |    |        |    |









17 JAN 37

24342

141635

Thesis  
L604  
c.1

Lew

A three dimensional  
solution of the tran-  
sient field problem  
using isoparametric  
finite elements.

17 JAN 37

24342

Thesis

141635

L604  
c.1

Lew

A three dimensional  
solution of the tran-  
sient field problem  
using isoparametric  
finite elements.

thesL604

A three dimensional solution of the tran



3 2768 002 11866 3

DUDLEY KNOX LIBRARY